МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

І. А. Дичка, В. П. Легеза, М. В. Онай

КОМП'ЮТЕРНА ЛОГІКА. ПРИКЛАДНА ТЕОРІЯ ЦИФРОВИХ АВТОМАТІВ: КОМП'ЮТЕРНИЙ ПРАКТИКУМ

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського як навчальний посібник для студентів, які навчаються за спеціальністю 121 «Інженерія програмного забезпечення», спеціалізацією «Програмне забезпечення комп'ютерних та інформаційно-пошукових систем»

Київ
КПІ ім. Ігоря Сікорського
2018

Рецензент *Романкевич В. О.*, канд. техн. наук, доц.

Відповідальний редактор *Сулема Є. С.*, канд. техн. наук, доц.

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського
(протокол № 7 від 29.03.2018 р.)
за поданням Вченої ради факультету прикладної математики
(протокол № 8 від 26.03.2018 р.)

Електронне мережне навчальне видання

Дичка Іван Андрійович, д-р техн. наук, проф.
Легеза Віктор Петрович, д-р техн. наук, проф.
Онай Микола Володимирович, канд. техн. наук, старший викладач

КОМП'ЮТЕРНА ЛОГІКА. ПРИКЛАДНА ТЕОРІЯ ЦИФРОВИХ АВТОМАТІВ: КОМП'ЮТЕРНИЙ ПРАКТИКУМ

Комп'ютерна логіка. Прикладна теорія цифрових автоматів: комп'ютерний практикум [Електронний ресурс] : навч. посіб. для студ. спеціальності 121 «Інженерія програмного забезпечення», спеціалізації «Програмне забезпечення комп'ютерних та інформаційно-пошукових систем» / І. А. Дичка, В. П. Легеза, М. В. Онай ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 3,85 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2018. – 88с.

Навчальний посібник розроблено для ознайомлення студентів з методиками проектування та дослідження вузлів і пристроїв комп'ютерних систем, а також вимогами до виконання лабораторних робіт, зокрема правилами їх оформлення. Навчальне видання призначене для студентів, які навчаються за спеціальністю 121 Інженерія програмного забезпечення, спеціалізацією «Програмне забезпечення комп'ютерних та інформаційно-пошукових систем» факультету прикладної математики КПІ ім. Ігоря Сікорського.

ЗАГАЛЬНІ ВКАЗІВКИ ДО ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ

Виконання лабораторних робіт з дисципліни «Комп'ютерна логіка» дозволяє закріпити теоретичні знання, оволодіти методиками проектування та дослідження вузлів і пристроїв комп'ютерних систем.

Кожній лабораторній роботі має передувати самостійна підготовка студентів, у процесі якої вони вивчають теоретичний матеріал, що стосується виконуваної роботи, та відповідну літературу. У процесі підготовки студент зобов'язаний скласти протокол лабораторної роботи, в якому мають знайти відображення всі пункти завдання (схеми, таблиці, часові діаграми тощо).

Студент, який не має протоколу, до виконання лабораторної роботи не допускається. Перед початком роботи студент повинен сформулювати мету і порядок виконання лабораторної роботи та відповісти на контрольні запитання викладача.

Під час виконання роботи студент зобов'язаний неухильно дотримуватись правил техніки безпеки і точно витримувати порядок проведення експериментів.

Перед початком наступного заняття в лабораторії студент зобов'язаний подати викладачеві повністю оформлений звіт з попередньої лабораторної роботи. Звіт має містити всі необхідні схеми, формули, таблиці, діаграми, графіки, одержані у процесі експериментального дослідження схем, а також підсумкові висновки.

Студент, який не подав звіту, до виконання наступної лабораторної роботи не допускається.

Студент одержує залік з лабораторної роботи після співбесіди з викладачем за тематикою виконаної роботи.

Для виконання лабораторних робіт використовується спеціальне програмне забезпечення – програмний комплекс для моделювання логічних схем.

Програмний комплекс для моделювання логічних схем

Програмний комплекс для моделювання логічних схем ПРОГМОЛС 2.0 (AFDK 2.0 – Advanced Functional Designer's Kit) призначений для моделювання процесів у комбінаційних схемах та схемах з пам'яттю. Він дозволяє створювати та редагувати логічні схеми, здійснювати моделювання їх роботи у синхронному (без урахування затримок сигналів у елементах схеми) та асинхронному (з урахуванням за-

тримок) режимах, а також зберігати отримані моделі (рис.1) у вигляді файлів на дисках.

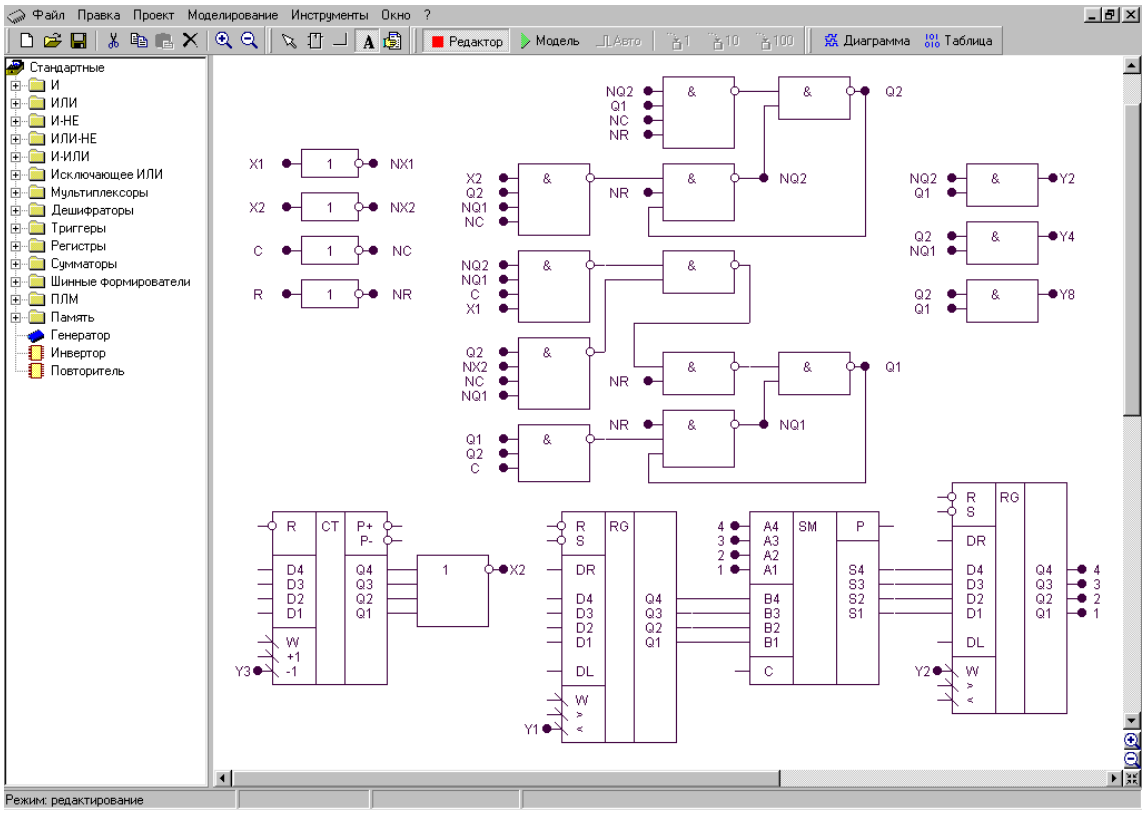


Рис.1. Зображення моделі операційного пристрою

Комплекс включає систему підказок, що полегшує роботу в різних режимах моделювання. Для роботи з програмою використовується система ієрархічних меню.

Меню містить наступні розділи:

- *Файл;*
- *Виправлення (Правка);*
- *Проект;*
- *Моделювання;*
- *Інструменти;*
- *Вікно;*
- *Допомога (?).*

Меню *Файл*

Меню *Файл* включає стандартні команди для керування файлами проектів.

Створити (комбінація клавіш **Ctrl+N)**. Створює новий файл проекту.

Відкрити (Ctrl +O). Викликає діалогове вікно відкривання проекту.

Зберегти (Ctrl +S). Зберігає файл проекту на диску під поточним ім'ям.

Зберегти як (Ctrl +A). Зберігає файл проекту на диску і запитує ім'я файлу та шлях до нього.

Закрити (Ctrl +F4). Закриває поточний файл проекту.

Вихід (Alt+F4). Закриває програму і всі вікна.

Меню *Виправлення (Правка)*

Меню *Виправлення (Правка)* включає стандартні команди редактора логічних схем.

Вирізати (Ctrl+X). Вирізає фрагмент схеми у внутрішній буфер обміну програми.

Копіювати (Ctrl+C). Копіює фрагмент схеми у внутрішній буфер обміну програми.

Вставити (Ctrl+V). Вставляє фрагмент у внутрішній буфер обміну програми.

Вилучити (Delete). Вилучає фрагмент схеми.

Меню *Проект*

Меню *Проект* включає команди керування проектом.

Корпус мікросхеми. Показати/приховати корпус мікросхеми поточного проекту.

Редактор корпусу. Викликає діалогове вікно редактора корпусу мікросхеми поточного проекту.

Компіляція (Alt+C). Компілює поточний проект. Дана команда використовується для відновлення списку змінних у діаграмі і таблиці проекту після зміни схеми без включення режиму моде-

лювання. При включенні режиму моделювання компіляція здійснюється автоматично.

Додати до бібліотеки. Копіює поточний проект у буфер обміну редактора бібліотек і викликає редактор бібліотек. Для вставки компонента в бібліотеку необхідно викликати команду **Вставити** редактора бібліотек.

Настроювання (Ctrl+F4). Викликає діалогове вікно настроювання проекту.

Меню *Моделювання*

Меню *Моделювання* включає команди для моделювання поточного проекту.

Крок (S). Відпрацювати крок модельного часу. Модельний час, що відповідає кроку, вибирається на сторінці **Модель** *Діалогового вікна* настроювань програми.

Відпрацювати інтервал. Відпрацьовує заданий користувачем інтервал модельного часу.

Відпрацювати до. Відпрацьовує до моменту модельного часу, заданого користувачем.

Генератор (G). У синхронному режимі викликає чергову зміну стану генератора і відпрацьовує схему до закінчення перехідних процесів. В асинхронному режимі відпрацьовує 1 такт модельного часу.

Меню *Інструменти*

Меню *Інструменти* включає команди виклику інструментів.

Редактор бібліотек (Alt+L). Активізує редактор бібліотек.

Настроювання. Викликає діалогове вікно настроювань програми.

Діаграма (Alt+D). Виводить на екран діаграму станів змінних поточного проекту.

Таблиця (Alt+T). Виводить на екран таблицю станів змінних поточного проекту.

Меню *Вікно*

Меню *Вікно* містить стандартні для MDI-інтерфейсу команди маніпуляції відкритих проектів.

Каскадом. Розташовує вікна відкритих проектів каскадом.

Розділити по вертикалі. Розташовує вікна відкритих проектів по вертикалі таким чином, щоб вони не перекривалися.

Розділити по горизонталі. Розташовує вікна відкритих проектів по горизонталі таким чином, щоб вони не перекривалися.

Збільшити. Збільшує масштаб схеми поточного проекту.

Зменшити. Зменшує масштаб схеми поточного проекту.

Меню *Допомога (?)*

Меню *Допомога (?)* містить наступні команди.

Про програму. Виводить відомості про програму та розробників.

Допомога (?). Викликає файл довідки.

Діалогові вікна

Діалогове вікно настроювання проекту.
Діалогове вікно настроювання програми.

Панель інструментів

Меню містить наступні панелі:

Стандартна

Панель *Стандартна* містить стандартні кнопки редакторів


Створити (Ctrl+N). Створює новий файл проекту.


Відкрити (Ctrl+O). Викликає діалогове вікно відкривання проекту.


Зберегти (Ctrl+S). Зберігає файл проекту на диску під поточним ім'ям.


Вирізати (Ctrl+X). Вирізає фрагмент схеми у внутрішній буфер обміну програми.


Копіювати (Ctrl+C). Копіює фрагмент схеми у внутрішній буфер обміну програми.


Вставити (Ctrl+V). Вставляє фрагмент схеми **з** внутрішнього буфера обміну програми.


Вилучити (Delete). Вилучає фрагмент схеми.


Збільшити. Збільшує масштаб схеми поточного проекту.


Зменшити. Зменшує масштаб схеми поточного проекту.

Редактор

Панель *Правка* містить інструменти редактора логічних схем:

Виділити. Дозволяє виділити і перетягнути лівою кнопкою миші фрагмент схеми, а також інвертувати виходи та входи елемента подвійним натисканням.

Елемент. Дозволяє вставити новий елемент у схему.

Лінія. Дозволяє провести зв'язок у схемі.

Змінна. Дозволяє задати змінну на схемі.

Захоплення. Дозволяє перетягувати лівою кнопкою миші весь зміст вікна редактора логічних схем.

Моделювання

Панель *Редагування* дозволяє керувати режимами редактора логічних схем і режимами моделювання.

Редактор. Переводить редактор логічних схем у режим редагування.

Модель. Переводить редактор логічних схем у режим моделювання.

Авто. У натиснутому стані вмикає режим автоматичного моделювання, а у відпущеному – режим покрокового моделювання.

1 такт. Відпрацьовує 1 такт модельного часу.

10 тактів. Відпрацьовує 10 тактів модельного часу.

100 тактів. Відпрацьовує 100 тактів модельного часу.

Інструменти

Панель *Інструменти* включає команди виклику інструментів.

Діаграма (Alt+D). Виводить на екран діаграму станів змінних поточного проекту.

Таблиця (Alt+T). Виводить на екран таблицю станів змінних поточного проекту.

Панель компонентів

Панель компонентів розташована в лівій частині екрана дисплея і є ієрархічним списком, у якому відображені під'єднані до програми бібліотеки та їх зміст. Кожна бібліотека містить ієрархічну структуру компонентів подібну до файлової системи. Панель компонентів призначена для вибору компонента для подальшої його вставки в схему.

Для дослідження схем у загальному випадку необхідно виконати таку послідовність дій.

1. Створити за допомогою редактора логічну схему на екрані дисплея.
2. Позначити на схемі вхідні та вихідні змінні.
3. Створити заголовок таблиці істинності (перелічити вхідні і вихідні змінні) та сформувати послідовність вхідних наборів.
4. Задати необхідні величини затримок сигналів для елементів схеми.
5. Встановити початковий стан схеми.
6. Перейти до режиму моделювання схеми.

ЛАБОРАТОРНА РОБОТА №1. ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ КОМБІНАЦІЙНИХ СХЕМ

Мета роботи: опанувати методику проектування комбінаційних схем у заданому елементному базисі та дослідження їх характеристик.

Теоретичні відомості

Будь-яку схему (вузол, пристрій) можна вважати перетворювачем інформації, на вхід якої надходять двійкові послідовності, а значення на кожному виході є функцією від вхідних двійкових послідовностей.

Фізичну схему можна реалізувати за допомогою логічних елементів.

Логічний елемент – це електронна схема, яка реалізує певну перемикальну функцію. Сукупність логічних елементів, призначену для перетворення двійкових змінних, називають логічною схемою.

Якщо сукупність вихідних сигналів логічної схеми з n входами і m виходами в даний момент часу повністю визначається сукупністю вхідних сигналів у цей самий момент часу і не залежить від вхідних сигналів, що діяли у попередні моменти часу, то таку логічну схему називають комбінаційною схемою (КС). Вважають, що КС має один стан. Поведінку КС можна описати системою перемикальних функцій.

Розрізняють задачу аналізу та задачу синтезу комбінаційної схеми.

Сутність задачі аналізу комбінаційної схеми за її відомою структурою полягає у знаходженні системи перемикальних (булевих) функцій, яка описує поведінку (логіку роботи) цієї схеми.

Задача синтезу комбінаційної схеми полягає у побудові із заданого набору логічних елементів оптимальної комбінаційної схеми, яка реалізує задану систему перемикальних функцій.

Задача синтезу КС є оберненою до задачі аналізу КС.

Реалізація булевої функції на основі заданих логічних елементів

Процес синтезу КС з n входами і одним виходом поділяють на три етапи.

На першому етапі, виходячи з таблиці істинності заданої перемикальної функції, яка описує роботу синтезованої КС, та беручи до уваги заданий елементний базис (типи логічних елементів), знаходять відповідну мінімальну нормальну форму функції. Якщо задана функція є неповністю визначеною (задана не на всіх 2^n наборах), то попередньо задану функцію довизначають у такий спосіб, щоб її нормальна форма була мінімальною.

На другому етапі, беручи до уваги кількість входів логічних елементів, заданих для синтезу КС, функцію записують в операторній формі, тоб-

то у вигляді суперпозиції операторів логічних елементів. Оператор логічного елемента – це функція, яку реалізує логічний елемент з урахуванням кількості його входів. Наприклад, оператором двовходового елемента І-НЕ є $\overline{x_2 x_1}$.

На третьому етапі, виходячи з операторної форми (операторного подання) функції, будують комбінаційну схему.

Розглянемо синтез КС в елементному базисі {І, АБО, НЕ, І-НЕ, АБО-НЕ}.

Нехай деяка функція $y_1 = f(x_3, x_2, x_1)$ задана таблицею істинності (табл. 1.1), якій відповідає діаграма Вейча на рис. 1.1.

Таблиця 1.1. Таблиця істинності функції y_1

x_3	x_2	x_1	y_1
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

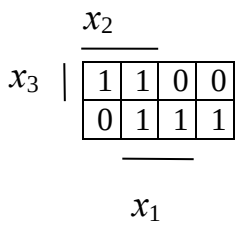


Рис 1.1. Діаграма Вейча функції y_1

МДНФ цієї функції має вигляд

$y_1 = f(x_3, x_2, x_1) = x_3x_2 \vee \overline{x}_3x_1 \vee \overline{x}_3\overline{x}_2$.

Послідовно дістанемо можливі канонічні нормальні форми цієї функції з позначенням внутрішньої та зовнішньої функції. Наприклад, у МДНФ функції y_1 внутрішньою є функція І, а зовнішньою – АБО.

$$y_1 = f(x_3, x_2, x_1) = x_3x_2 \vee \overline{x}_3x_1 \vee \overline{x}_3\overline{x}_2 = \quad \text{(форма І / АБО)}$$

$$= \overline{\overline{x_3x_2 \vee \overline{x}_3x_1 \vee \overline{x}_3\overline{x}_2}} =$$

$$= \overline{\overline{x_3x_2} \,\&\, \overline{\overline{x}_3x_1} \,\&\, \overline{\overline{x}_3\overline{x}_2}} = \quad \text{(форма І-НЕ / І-НЕ)}$$

$$= \overline{(\overline{x}_3 \vee \overline{x}_2)(x_3 \vee \overline{x}_1)(x_3 \vee x_2)} = \quad \text{(форма АБО / І-НЕ)}$$

$$= \overline{\overline{x}_3 \vee \overline{x}_2} \vee \overline{x_3 \vee \overline{x}_1} \vee \overline{x_3 \vee x_2} \quad \text{(форма АБО-НЕ / АБО).}$$

Скориставшись МДНФ заперечення заданої функції, тобто
$\overline{y}_1 = \overline{f}(x_3, x_2, x_1) = x_3\overline{x}_2 \vee \overline{x}_3 x_2 \overline{x}_1$ (див. діаграму Вейча)
дістанемо ще чотири канонічні нормальні форми :

$$y_1 = f(x_3, x_2, x_1) = \overline{x_3\overline{x}_2 \vee \overline{x}_3 x_2 \overline{x}_1} = \quad \text{(форма І / АБО-НЕ)}$$
$$= \overline{x_3\overline{x}_2}\ \&\ \overline{\overline{x}_3 x_2 \overline{x}_1} = \quad \text{(форма І-НЕ / І)}$$
$$= (\overline{x}_3 \vee x_2)(x_3 \vee \overline{x}_2 \vee x_1) = \quad \text{(форма АБО / І)}$$
$$= \overline{\overline{(\overline{x}_3 \vee x_2)(x_3 \vee \overline{x}_2 \vee x_1)}} =$$
$$= \overline{\overline{x}_3 \vee x_2} \vee \overline{x_3 \vee \overline{x}_2 \vee x_1} \quad \text{(форма АБО-НЕ / АБО-НЕ).}$$

Усі одержані форми є дворівневими (нормальними), і, якщо логічні елементи, які застосовують для побудови КС, мають не менше трьох входів, то відповідні КС будують двокаскадними.

Якщо ж кількість входів p у логічних елементів, що використовуються для побудови КС, менша, ніж це необхідно для реалізації одержаної нормальної форми, то змінні об'єднують у групи так, щоб до кожної групи входило не більше p змінних, застосовуючи при цьому такі співвідношення:

$x_n \cdot x_{n-1} \cdot \ldots \cdot x_1 = (x_n \cdot \ldots \cdot x_{n-g+1}) \cdot \ldots \cdot (x_s \cdot \ldots \cdot x_1)$;
$x_n \vee x_{n-1} \vee \ldots \vee x_1 = (x_n \vee \ldots \vee x_{n-g+1}) \vee \ldots \vee (x_s \vee \ldots \vee x_1)$;
$\overline{x_n \cdot x_{n-1} \cdot \ldots \cdot x_1} = \overline{(x_n \cdot \ldots \cdot x_{n-g+1})} \cdot \ldots \cdot \overline{(x_s \cdot \ldots \cdot x_1)}$;
$\overline{x_n \vee x_{n-1} \vee \ldots \vee x_1} = \overline{(x_n \vee \ldots \vee x_{n-g+1})} \vee \ldots \vee \overline{(x_s \vee \ldots \vee x_1)}$, де $g, s <= p$.

Кількість груп змінних також не має перевищувати p, інакше вказані перетворення виконують стосовно груп змінних.

Застосування вказаних співвідношень дозволяє подати задану перемикальну функцію в операторній формі з урахуванням кількості входів логічних елементів, якими належить скористатися під час побудови КС. Але одержана в цьому випадку форма функції не є нормальною (дворівневою), і, відповідно, КС буде мати більше, ніж два каскади логічних елементів.

Нехай для реалізації функції y_1 необхідно скористатися двовходовими логічними елементами І та І-НЕ, тобто елементами 2І та 2І-НЕ.

У цьому випадку слід використати форму І-НЕ / І-НЕ або форму І-НЕ / І функції. Наприклад, нехай форма І-НЕ / І функції y_1 має вигляд

$$y_1 = f(x_3, x_2, x_1) = \overline{x_3\overline{x}_2} \cdot \overline{\overline{x}_3 x_2 \overline{x}_1}\,.$$

Операторна форма цієї функції для випадку заданих логічних елементів виглядає так:

$$\overline{x_3\overline{x}_2} \cdot \overline{\overline{x}_3 x_2 \overline{x}_1} = \overline{x_3\overline{x}_2} \cdot \overline{\overline{\overline{x}_3 x_2} \cdot \overline{\overline{x}_1}} = \overline{x_3\overline{x}_2} \cdot \overline{\overline{\overline{x}_3 x_2} \cdot \overline{x}_1}$$

Одержана форма функції є чотирирівневою. Їй відповідає чотирикаскадна КС на рис. 1.2.

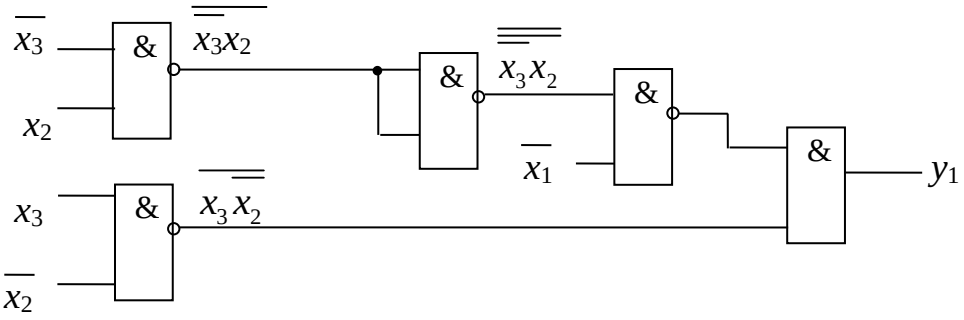


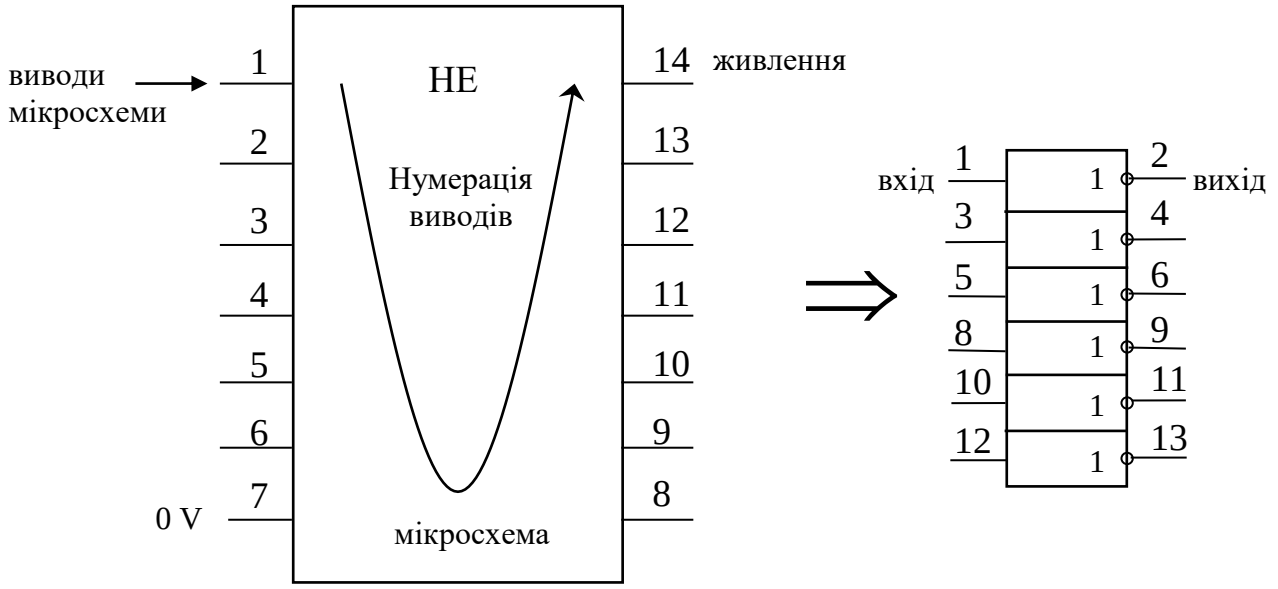
Рис. 1.2. КС, що реалізує перемикальну функцію y_1

У загальному випадку заданій системі логічних елементів може відповідати декілька операторних форм функції, а, отже, можна побудувати декілька КС, що реалізують одну й ту саму булеву функцію.

Наприклад, якщо до складу заданої системи елементів входять логічні елементи І, АБО та І-НЕ, то для одержання операторної форми функції можна використати як вихідну одну з п’яти нормальних форм: І / АБО, І-НЕ / І-НЕ, АБО / І-НЕ, І-НЕ / І, АБО / І. Отже, можна дістати п’ять різних операторних форм функції й побудувати п’ять різних КС, які реалізують одну й ту саму перемикальну функцію.

Для вибору оптимальної КС з кількох можливих необхідно порівняти всі КС за заданими параметрами, наприклад, за складністю та/або швидкодією.

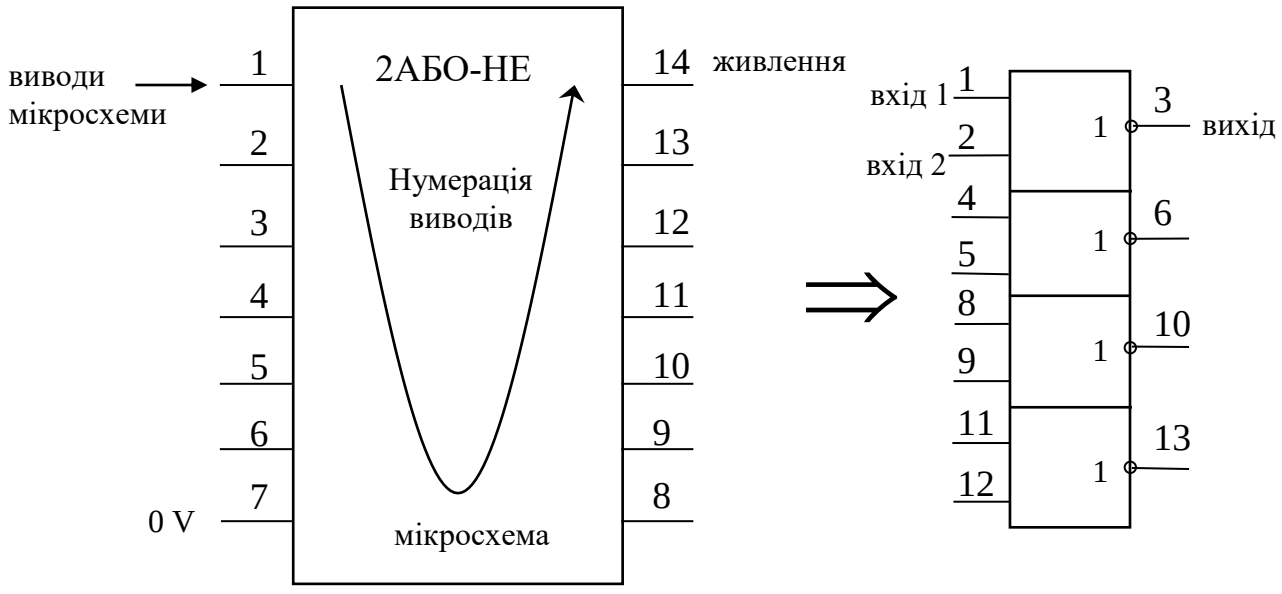
Фізично логічні елементи виготовляють лише в складі мікросхеми. Наприклад, якщо нам необхідно використати інвертор (англ. NOT) – найпростіший логічний елемент, то ми повинні розуміти, що інвертор є складовою частиною мікросхеми, яка містить 6 інверторів (рис. 1.3):



(14 виводів)

Рис. 1.3. Мікросхема, що містить інвертори

Або, наприклад, якщо нам необхідно використати 2-входовий логічний елемент АБО-НЕ, який позначається як 2АБО-НЕ (англ. 2NOR), то ми повинні розуміти, що елемент 2АБО-НЕ є складовою частиною мікросхеми, яка містить 4 елементи 2АБО-НЕ (рис. 1.4.):



(14 виводів)

Рис. 1.4. Мікросхема, що містить двовходові елементи АБО-НЕ

Складність КС оцінюють трьома способами:

1) способом Квайна (K) – складність КС дорівнює сумарній кількості входів усіх її логічних елементів;

2) за кількістю логічних елементів (M) – складність КС дорівнює загальній кількості логічних елементів, які входять до складу КС;

3) за кількістю умовних корпусів мікросхем (N) – складність КС дорівнює загальній кількості умовних корпусів мікросхем, на які можна розбити схему.

Кількість умовних корпусів мікросхем визначають за формулою:

$$N = \sum_{i=1}^{r} (V_i W_i / 14), \qquad (1)$$

де r – кількість типів мікросхем (типів логічних елементів); V_i – кількість мікросхем i-го типу, W_i – кількість виводів мікросхеми.

За умовний вважають корпус мікросхеми на 14 виводів.

Параметри K та M доцільно застосовувати при реалізації КС у вигляді інтегральної схеми, оскільки складність інтегральної схеми за заданої площі кристала пропорційна кількості логічних елементів та кількості їх входів.

Параметр N слід застосовувати при порівнянні складності КС, побудованих на мікросхемах.

Наприклад, складність КС на рис. 1.5, дорівнює:

- за першим способом (спосіб Квайна) $\rightarrow K = 8$ (сумарна кількість входів всіх логічних елементів);
- за другим способом $\rightarrow M = 5$ (загальна кількість логічних елементів);
- за третім способом $\rightarrow N = \left\lceil \frac{3_{\text{2АБО-НЕ}}}{4} \right\rceil + \left\lceil \frac{2_{\text{НЕ}}}{6} \right\rceil = 1_{\text{2АБО-НЕ}} + 1_{\text{НЕ}} = 2$ умовні корпуси (вважаємо, що одна мікросхема НЕ містить 6 інверторів, а одна мікросхема 2АБО-НЕ – 4 елементи 2АБО-НЕ).

Швидкодія КС залежить від часових параметрів логічних елементів t^{01} (час переходу вихідного сигналу з 0 в 1) та t^{10} (час переходу вихідного сигналу з 1 в 0), які характеризують затримку сигналу елементом. На практиці використовують середнє значення часу затримки $t = (t^{01} + t^{10}) / 2$ або максимальне $t^* = \max(t^{01}, t^{10})$.

Якщо КС побудована на однотипових логічних елементах, то середній час затримки T сигналу дорівнює $T = Lt$, де L – рівень схеми, який дорівнює кількості елементів, що входять в максимальний за довжиною ланцюжок елементів. Якщо ж у КС використовуються елементи з різною за-

тримкою сигналів, то у схемі визначають шлях, який потребує максимального часу поширення сигналу.

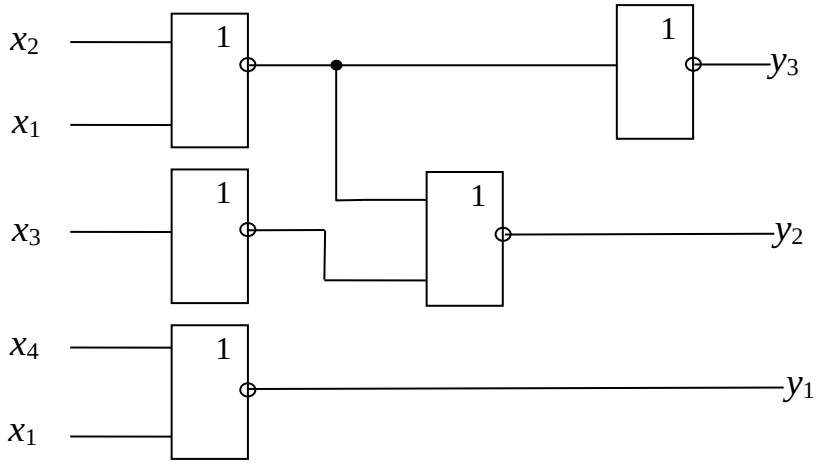


Рис.1.5. Приклад КС з чотирма входами і трьома виходами

Таким чином, з кількох можливих КС вибирають ту комбінаційну схему, яка якнайповніше задовольняє заданим параметрам.

Проектування комбінаційних схем з кількома виходами відрізняється від запропонованої методики тим, що кожну функцію, яка входить до системи перемикальних функцій, мінімізують окремо, а потім будують спільну КС, прагнучи, щоб кількість використаних логічних елементів була мінімальною, або вдаються до спільної мінімізації системи перемикальних функцій. Кінцевою метою при цьому є реалізація схеми за допомогою мінімальної кількості логічних елементів (корпусів мікросхем).

Під час реалізації перемикальних функцій в окремих випадках можна зменшити у схемі кількість логічних елементів (корпусів мікросхем), використовуючи, наприклад, дешифратор або мультиплексор.

Реалізація булевої функції з використанням мультиплексора

Застосування мультиплексора для реалізації перемикальної функції від n змінних ґрунтується на розкладанні функції за m змінними:

$$f(x_n, ..., x_1) = \overrightarrow{V}_{\alpha_m} x_n^{\alpha_n} \cdot x_{n-1}^{\alpha_{n-1}} \cdot ... \cdot x_{n-m+1}^{\alpha_{n-m+1}} \cdot f(\alpha_n, \alpha_{n-1}, ..., \alpha_{n-m+1}, x_{n-m}, x_{n-m-1}, ..., x_1),$$

де $x_i^{\alpha_i} = \begin{cases} \overline{x}_i, \text{ при } \alpha_i = 0, \\ x_i, \text{ при } \alpha_i = 1. \end{cases}$

$\bigvee\limits_{\vec{\alpha}_m}$ – диз'юнкція по всіх наборах $\vec{\alpha}_m = (\alpha_n, \alpha_{n-1}, ..., \alpha_{n-m+1})$.

Таке розкладання дозволяє виключити m з n змінних перемикальної функції, подавши її через інші функції, що залежать лише від n-m змінних.

Наприклад, розкладання функції $y_3 = f(x_n, ..., x_1)$ за змінними x_n та x_{n-1} має вигляд

$y_3 = \overline{x}_n\overline{x}_{n-1} f(0, 0, x_{n-2}, ..., x_1) \vee \overline{x}_n x_{n-1} f(0, 1, x_{n-2}, ..., x_1) \vee x_n\overline{x}_{n-1} f(1,0, x_{n-2}, ..., x_1)$
$\vee x_n x_{n-1} f(1, 1, x_{n-2}, ..., x_1) = \overline{x}_n\overline{x}_{n-1} f_0(x_{n-2}, ..., x_1) \vee \overline{x}_n x_{n-1} f_1(x_{n-2}, ..., x_1) \vee$
$\vee x_n\overline{x}_{n-1} f_2(x_{n-2}, ..., x_1) \vee x_n x_{n-1} f_3(x_{n-2}, ..., x_1)$, де f_0, f_1, f_2, f_3 – функції від $n-2$ змінних.

Такому поданню функції відповідає схема на рис. 1.6, де $f_0 - f_3$ – комбінаційні схеми, що реалізують відповідні функції.

Нехай задано функцію $y_4 = f(x_4, x_3, x_2, x_1)$ від чотирьох змінних, якій відповідає діаграма Вейча на рис. 1.7.

Розкладання y_4 за змінними x_4 та x_3 має вигляд

$y_4 = \overline{x}_4\overline{x}_3 f(0, 0, x_2, x_1) \vee \overline{x}_4 x_3 f(0, 1, x_2, x_1) \vee x_4\overline{x}_3 f(1, 0, x_2, x_1) \vee$
$\vee \quad x_4 x_3 f(1, 1, x_2, x_1) = \overline{x}_4\overline{x}_3 f_0(x_2, x_1) \vee \overline{x}_4 x_3 f_1(x_2, x_1) \vee$
$\vee \quad x_4\overline{x}_3 f_2(x_2, x_1) \vee x_4 x_3 f_3(x_2, x_1)$.

Для визначення функцій $f_0 - f_3$ діаграму Вейча функції y_4 можна розглядати як 4 самостійні діаграми, кожній з яких відповідає своя f_i ($i = 0 - 3$) (рис. 1.8).

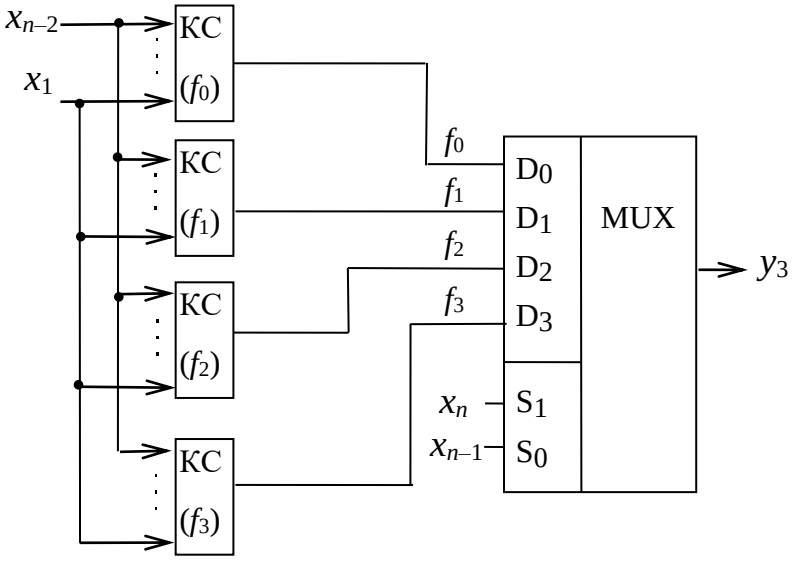


Рис. 1.6. Реалізація функції y_3 від n змінних за допомогою мультиплексора з двома входами керування

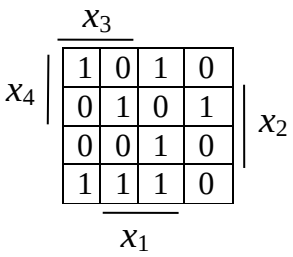


Рис. 1.7. Діаграма Вейча функції y_4

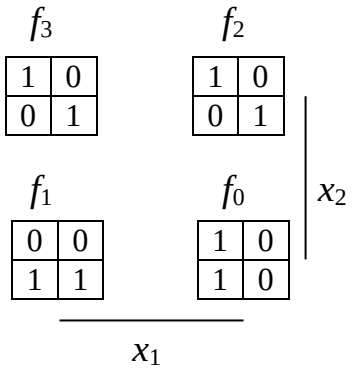


Рис. 1.8. Діаграми Вейча функцій $f_0 - f_3$ при розкладанні y_4 за змінними x_4 та x_3

Розкладання y_4 за x_4 та x_3 дає такі функції $f_0 - f_3$:

$f_0 = x_1$, $f_1 = \overline{x}_2$, $f_2 = \overline{x}_2 x_1 \vee x_2 \overline{x}_1$, $f_3 = \overline{x}_2 \overline{x}_1 \vee x_2 x_1$.

Якщо для реалізації функцій $f_0 - f_3$ необхідно застосувати, наприклад, 2-входові елементи І-НЕ, то $f_0 - f_3$ слід подати у формі І-НЕ / І-НЕ :

$f_0 = x_1$, $f_1 = \overline{x}_2$, $f_2 = \overline{\overline{\overline{x}_2 x_1} \,\&\, \overline{x_2 \overline{x}_1}}$, $f_3 = \overline{\overline{\overline{x}_2 \overline{x}_1} \,\&\, \overline{x_2 x_1}}$.

Комбінаційна схема, яка реалізує функцію y_4 за допомогою мультиплексора з двома входами керування, показана на рис. 1.9.

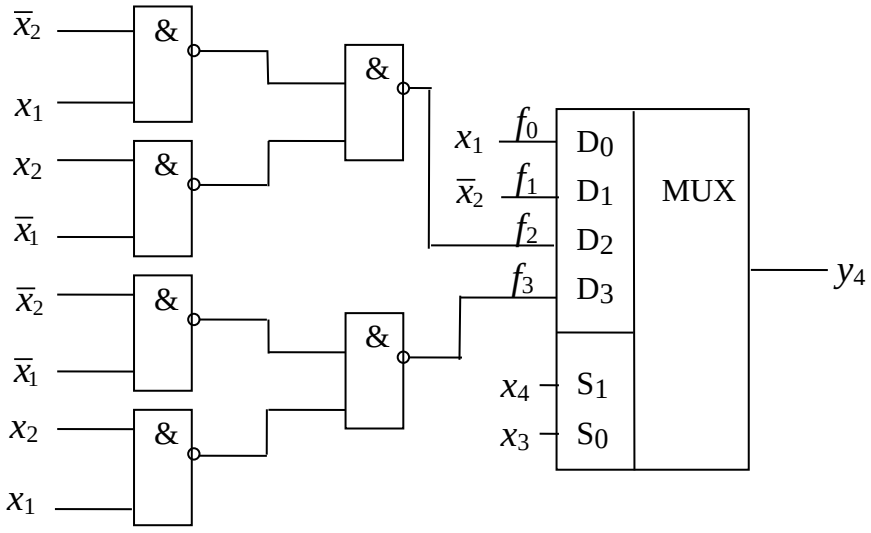


Рис. 1.9. Реалізація функції y_4 за допомогою мультиплексора (функції $f_0 - f_3$ відповідають розкладанню y_4 за змінними x_4 та x_3)

Але функцію y_4 можна розкласти також за змінними x_4, x_2; x_4, x_1; x_3, x_2; x_3, x_1; x_2, x_1. Складність схеми у кожному випадку може бути різною. Тому для одержання найпростішої схеми необхідно порівняти всі КС, кожній з яких відповідає своя пара змінних, за якими виконано розкладання функції y_4. На рис. 1.10 та 1.11 зображені решта 5 варіантів визначення функцій $(\gamma_0, \delta_0, \lambda_0, \eta_0, \omega_0) - (\gamma_3, \delta_3, \lambda_3, \eta_3, \omega_3)$, а на рис. 1.12 – 1.14 – їх реалізація на 2-входових елементах І–НЕ. Найпростішою є реалізація функції y_4 у випадку її розкладання за змінними x_3 та x_1 (рис. 1.15).

Таким чином, у разі реалізації перемикальної функції з використанням мультиплексора складність схеми залежить від того, за якими змінними виконано розкладання функції. Для одержання найпростішої схеми потрібен перебір всіх комбінацій змінних, за якими розкладають функцію.

Під час обчислення складності КС з використанням мультиплексора вважають, що в одному корпусі мікросхеми розміщуються два мультиплексори і така мікросхема має 16 виводів. Складність мікросхеми мультиплексора складає 16/14=1,14 умовних корпусів.

Оцінимо складність КС на рис. 1.9 за формулою (1):

$$N = \left\lceil \frac{6_{\text{2I-HE}}}{4} \right\rceil \cdot \frac{14}{14} + \left\lceil \frac{1_{\text{MS}}}{2} \right\rceil \cdot \frac{16}{14} = 2 + 1{,}14 = 3{,}14 \quad \text{умовних корпусів.}$$

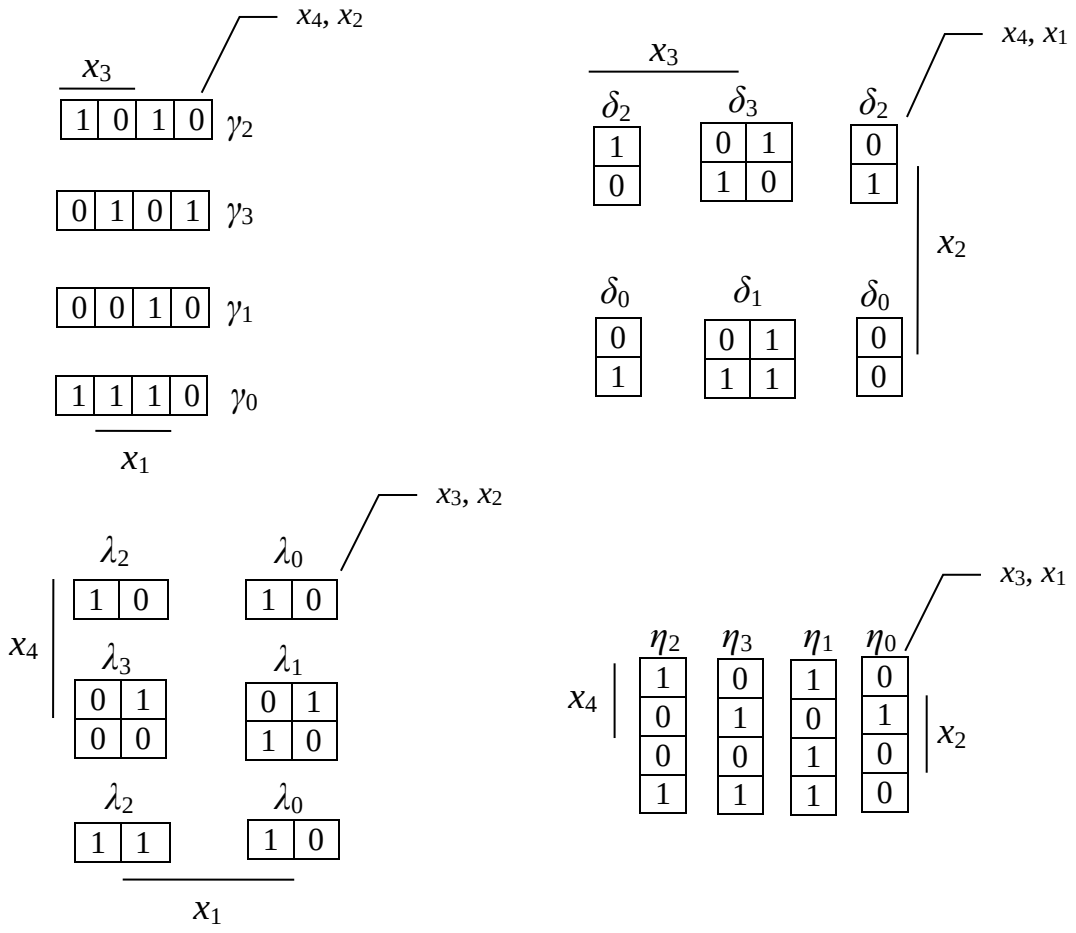


Рис. 1.10. Визначення функцій від двох змінних у випадку розкладання y_4 за змінними x_4, x_2; x_4, x_1; x_3, x_2; x_3, x_1

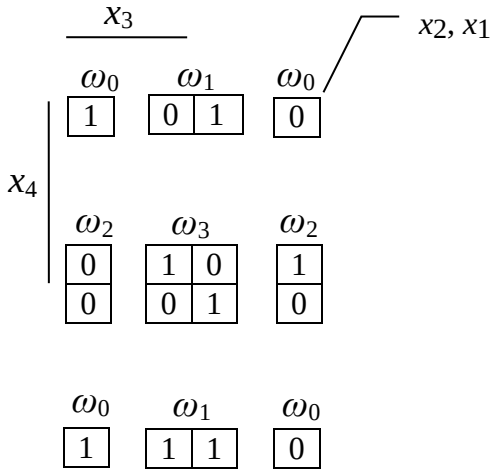


Рис. 1.11. Визначення функцій від двох змінних у випадку розкладання y_4 за змінними x_2, x_1

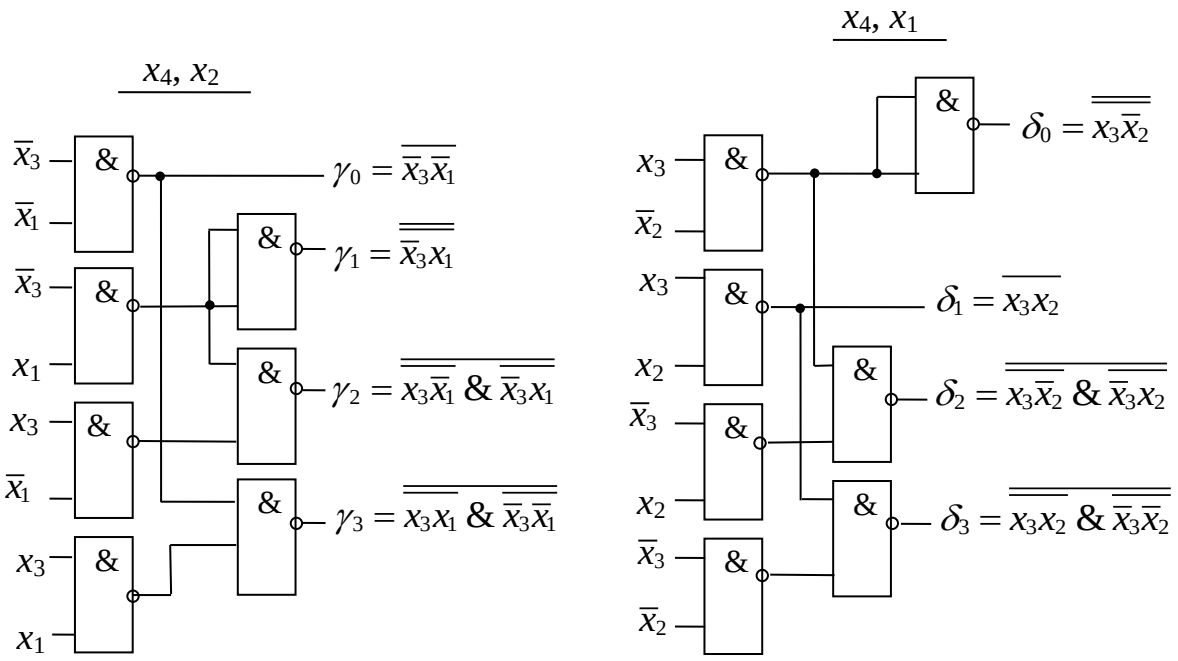


Рис. 1.12. Реалізація функцій від двох змінних у випадку розкладання y_4 за змінними x_4, x_2; x_4, x_1

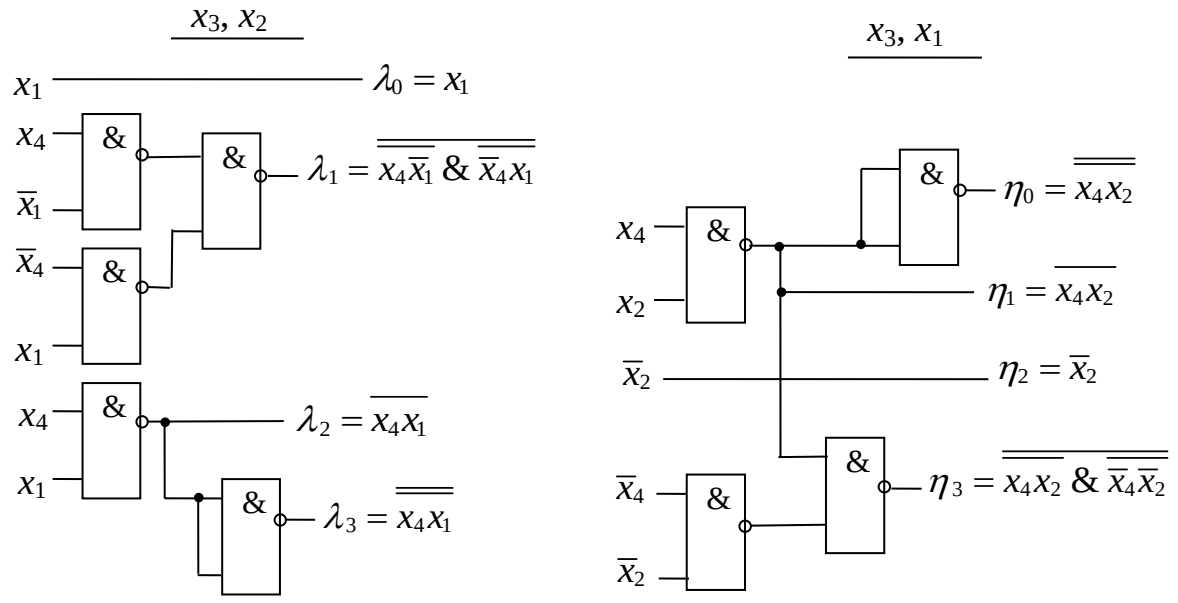


Рис. 1.13. Реалізація функцій від двох змінних у випадку розкладання y_4 за змінними x_3, x_2; x_3, x_1

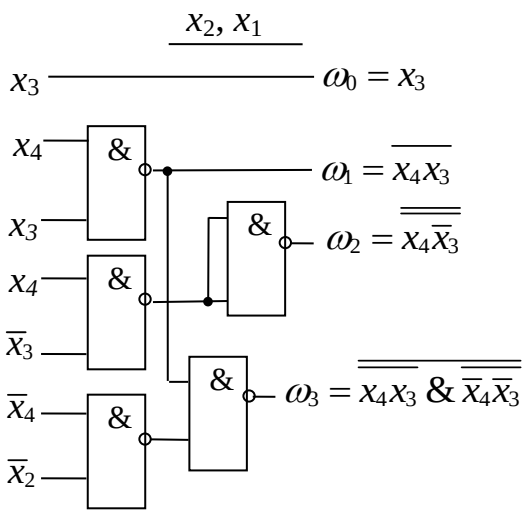


Рис. 1.14. Реалізація функцій від двох змінних у випадку розкладання y_4 за змінними x_2, x_1

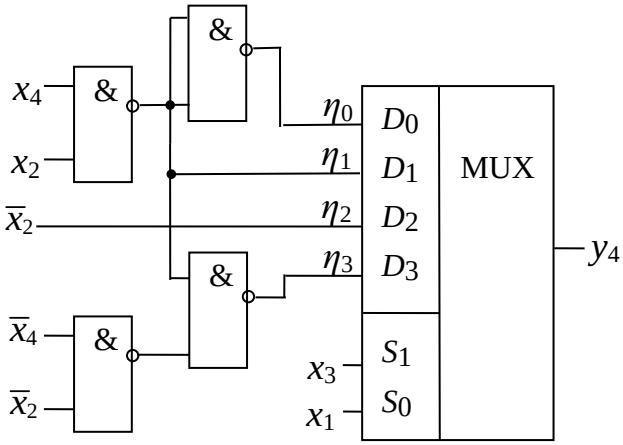


Рис. 1.15. Найоптимальніша реалізація функції y_4 за допомогою мультиплексора

Реалізація системи булевих функцій з використанням дешифратора

Оскільки дешифратор на r входів реалізує всі конституенти одиниці, то для реалізації перемикальної функції від r змінних достатньо за допомогою логічного елемента АБО одержати диз'юнкцію конституент одиниці тих наборів, на яких функція набуває значення 1. Якщо дешифратор має інверсні виходи, то замість елемента АБО слід використати елемент І-НЕ.

Наприклад, функцію

$y_2 = \overline{x}_3 x_2 x_1 \vee x_3 x_2 x_1 \vee \overline{x}_3 x_2 \overline{x}_1 = 3 \vee 7 \vee 2 = \overline{\overline{3 \vee 7 \vee 2}} = \overline{\overline{3} \cdot \overline{7} \cdot \overline{2}}$ можна реалізувати за допомогою 3-входового дешифратора з прямими (рис. 1.16 *а*) або інверсними (рис. 1.16 *б*) виходами.

Дешифратори доцільно застосовувати для реалізації систем перемикальних функцій (зокрема, для побудови перетворювачів кодів).

Наприклад, для реалізації перетворювача кодів, що має 4 входи (x_4, x_3, x_2, x_1) та 4 виходи (z_4, z_3, z_2, z_1) можна використати дешифратор на 4 входи з інверсними виходами. Такий дешифратор реалізовано у вигляді мікросхеми на 24 виводи. Складність такої мікросхеми становить 24/14 = 1,7 умовних корпусів.

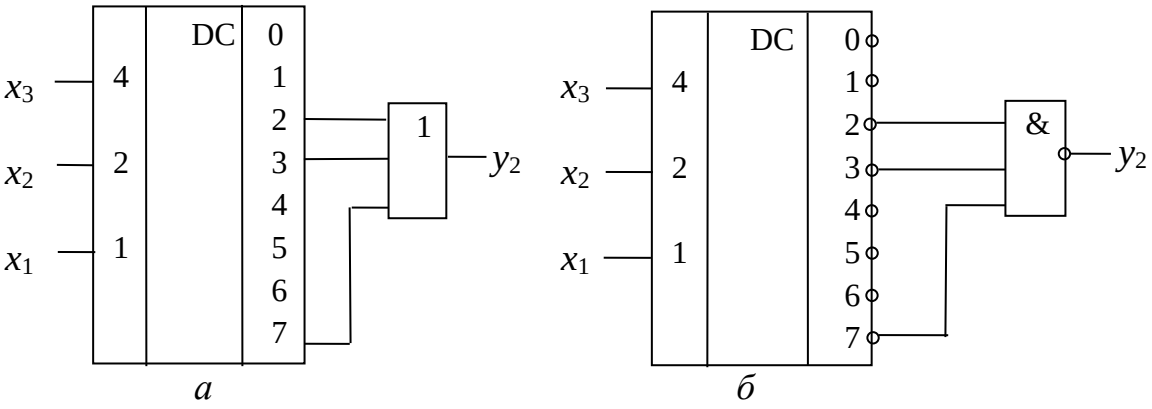


Рис. 1.16. Реалізація функції y_2 за допомогою дешифратора:
а – дешифратор з прямими виходами;
б – дешифратор з інверсними виходами

Кожен з виходів z_4, z_3, z_2, z_1 перетворювача вважають окремою функцією $z_i = f_i(x_4, x_3, x_2, x_1)$. Функцію z_i записують у вигляді диз'юнкції конституент одиниці тих наборів, на яких функція набуває значення 1. Для цього в таблиці істинності перетворювача кодів у стовпці z_i знаходять, на яких наборах функція z_i дорівнює 1, і далі записують диз'юнкцію цих наборів (набір подається у вигляді числа в діапазоні 0 – 15, де величина числа є порядковим номером набору (номером виходу дешифратора). Далі функцію z_i перетворюють у відповідну операторну форму (залежно від заданого типу логічних елементів та кількості їх входів). Для реалізації всіх z_i використовують один і той самий дешифратор.

Нехай перетворювач кодів задано таблицею істинності 1.3, а для його реалізації використовується дешифратор на 4 входи з інверсними виходами (кількість виходів – 16, їх нумерація – з 0 до 15) та 3-входові елементи І-НЕ.

Для реалізації першого виходу перетворювача (функції z_4) проглянемо стовпець z_4 у табл. 1.3. Функція z_4 дорівнює одиниці на таких наборах: 5, 6, 7, 8, 9, 10, 11, 12. Спочатку функцію z_4 запишемо у вигляді

$$z_4 = 5 \vee 6 \vee 7 \vee 8 \vee 9 \vee 10 \vee 11 \vee 12 = \overline{\overline{5 \vee 6 \vee 7 \vee 8 \vee 9 \vee 10 \vee 11 \vee 12}} =$$

$$= \overline{\bar{5} \cdot \bar{6} \cdot \bar{7} \cdot \bar{8} \cdot \bar{9} \cdot \overline{10} \cdot \overline{11} \cdot \overline{12}},$$

де, наприклад, $\bar{5}$ означає п'ятий вихід дешифратора (дешифратор має інверсні виходи). Далі z_4 подаємо у відповідній операторній формі (використовуються 3-входові елементи І-НЕ)

$$z_4 = \overline{\bar{5} \cdot \bar{6} \cdot \bar{7} \cdot \bar{8} \cdot \bar{9} \cdot \overline{10} \cdot \overline{11} \cdot \overline{12}} = \overline{\overline{\bar{5} \cdot \bar{6} \cdot \bar{7}} \cdot \overline{\bar{8} \cdot \bar{9} \cdot \overline{10}} \cdot \overline{\overline{11} \cdot \overline{12}}},$$

придатній для реалізації на логічних елементах 3І-НЕ.

Аналогічно реалізують функції z_3, z_2, z_1, але при цьому використовують виходи того самого дешифратора.

Таблиця 1.2. Таблиця істинності перемикальної функції *y*

x_4	x_3	x_2	x_1	y
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	α_1
0	1	0	0	α_2
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	α_3
1	0	0	1	0
1	0	1	0	α_4
1	0	1	1	1
1	1	0	0	α_5
1	1	0	1	0
1	1	1	0	1
1	1	1	1	α_6

Таблиця 1.3. Таблиця істинності перетворювача кодів

Входи				Виходи			
x_4	x_3	x_2	x_1	z_4	z_3	z_2	z_1
0	0	0	0	0	0	1	0
0	0	0	1	0	1	0	α_1
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	α_2
0	1	1	0	1	0	0	0
0	1	1	1	1	0	1	1
1	0	0	0	1	0	1	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	0	α_3
1	0	1	1	1	1	1	0
1	1	0	0	1	1	1	α_4
1	1	0	1	0	0	0	0
1	1	1	0	0	0	0	α_5
1	1	1	1	0	0	1	α_6

Завдання на лабораторну роботу

1. Довизначити перемикальну функцію, задану табл. 1.2. Для цього варіант (дві молодші цифри номера залікової книжки) подати у двійковій системі числення у вигляді слова $\alpha_6\ \alpha_5\ \alpha_4\ \alpha_3\ \alpha_2\ \alpha_1$. Значення α_i (i = 1 – 6) підставити в табл. 1.2.

Наприклад, якщо номер варіанта дорівнює 21 (010101), то $\alpha_6 = 0$, $\alpha_5 = 1$, $\alpha_4 = 0$, $\alpha_3 = 1$, $\alpha_2 = 0$, $\alpha_1 = 1$.

2. Подати функцію у восьми канонічних формах.

3. Одержати операторні форми функції, які можна реалізувати на логічних елементах, заданих табл. 1.4 (згідно з варіантом), і побудувати відповідні комбінаційні схеми, вважаючи, що на входи КС можуть надходити прямі та інверсні значення змінних.

4. Для кожної побудованої КС визначити її швидкодію (параметр T) та складність (параметр N – кількість умовних корпусів). Усі мікросхеми з табл. 1.4 мають по 14 виводів. Вибрати КС з максимальною швидкодією та мінімальними апаратними витратами.

5. Побудувати найоптимальнішу КС для реалізації функції, заданої табл. 1.2, використавши мультиплексор з двома входами керування та логічні елементи з табл. 1.4 (відповідно до варіанту). Порівняти складність одержаної схеми зі схемою, побудованою без використання мультиплексора, вважаючи, що два мультиплексори розміщуються в одному корпусі мікросхеми з 16-ма виводами.

6. Побудувати перетворювач кодів (табл. 1.3) , використавши елементи 3І-НЕ та 4-входовий дешифратор з інверсними виходами. Визначити значення складності *N* перетворювача, беручи до уваги, що дешифратор реалізовано у вигляді мікросхеми на 24 виводи.

Таблиця 1.4. Таблиця логічних елементів

α_3 α_2 α_1	Тип елементів	Кількість елементів у корпусі мікросхеми	Час затримки сигналу, t , нс
0 0 0	3І-НЕ 3І	3 3	20 24
0 0 1	4І-НЕ 2АБО	2 4	20 22
0 1 0	4І 2АБО	2 4	24 22
0 1 1	3І 2АБО	3 4	24 22
1 0 0	2АБО-НЕ 4І	4 2	22 24
1 0 1	2І-НЕ 2АБО	4 4	20 22
1 1 0	2АБО-НЕ 3І	4 3	20 24
1 1 1	2І-НЕ 2АБО	4 4	20 22

Порядок виконання роботи

1. За завданням викладача побудувати модель комбінаційної схеми, що реалізує функцію, задану табл. 1.2, використовуючи програмний комплекс ПРОГМОЛС 2.0.
2. Перевірити правильність роботи моделі комбінаційної схеми.
3. За завданням викладача побудувати модель КС з використанням мультиплексора та дослідити її роботу.

4. Побудувати модель перетворювача кодів, заданого табл. 1.3, та перевірити правильність його функціонування.

Вимоги до оформлення звіту

Звіт має включати:

1) синтез двох КС, які реалізують функцію, задану табл. 1.2, на логічних елементах, заданих табл. 1.4;

2) синтез шести КС, які реалізують функцію, задану табл. 1.2, з використанням мультиплексора та заданих логічних елементів;

3) схему перетворювача кодів, заданого табл. 1.3;

4) розрахунок показників складності та швидкодії для дев'яти вищеперелічених комбінаційних схем.

Питання для самоперевірки

1. Дати визначення комбінаційної схеми.
2. У чому полягає сутність задачі аналізу та задачі синтезу комбінаційної схеми?
3. Охарактеризувати основні етапи синтезу комбінаційної схеми на заданих логічних елементах.
4. Записати задану функцію у восьми канонічних нормальних формах.
5. Що таке операторна форма перемикальної функції?
6. Які співвідношення використовують для подання перемикальної функції в операторній формі?
7. Чим відрізняється операторна форма булевої функції від канонічної форми?
8. Як оцінюють складність комбінаційних схем?
9. Як визначити швидкодію комбінаційної схеми?
10. У чому полягає особливість синтезу комбінаційних схем з кількома виходами?
11. На чому ґрунтується застосування мультиплексора для реалізації перемикальної функції?

Рекомендована література

1. Жабін В.І., Жуков І.А., Клименко І.А., Ткаченко В.В. Прикладна теорія цифрових автоматів. – К. : Вид-во Нац. авіац. ун-ту "НАУ-друк", 2009. – 360 с.
2. Проектирование цифровых вычислительных машин: Учеб. пособие для студентов вузов / Под ред. С.А. Майорова. – М. : Высшая шк., 1972. – 344 с.
3. Бабич М.П., Жуков І.А. Комп'ютерна схемотехніка : Навч. посіб. – К. : МК-Прес, 2004. – 412 с.

ЛАБОРАТОРНА РОБОТА №2. ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ ТРИГЕРІВ НА ПОТЕНЦІАЛЬНИХ ЕЛЕМЕНТАХ

Мета роботи: вивчити закони функціонування тригерів різних типів і опанувати методику проектування та дослідження тригерів на потенціальних елементах.

Теоретичні відомості

Тригером називають пристрій, побудований на логічних елементах, який має два стійкі стани.

Тригерний пристрій (тригер) складається з бістабільної схеми (БС), яку часто називають запам'ятовувальним елементом, та схеми керування (СК) (рис. 2.1). СК є комбінаційною схемою з двома виходами. На рис. 2.1 використовуються такі позначення:

$x_n, \ldots, x_1$ – інформаційні входи тригера;

$C_m, \ldots, C_1$ – тактові входи (входи синхронізації);

Q – прямий вихід тригера;

$\overline{Q}$ – інверсний вихід тригера;

f_2, f_1 – функції збудження бістабільної схеми.

Зворотні зв′язки, показані пунктиром, можуть бути відсутні.

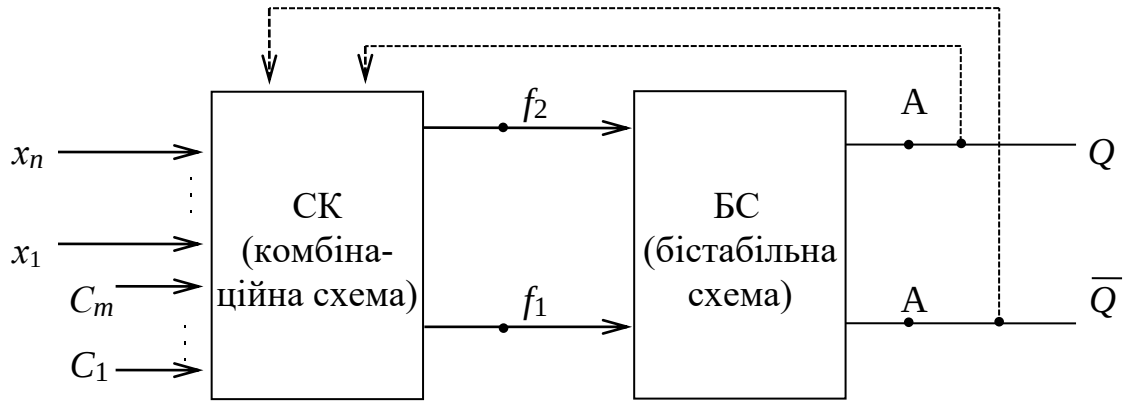


Рис. 2.1. Узагальнена структура тригера

Для забезпечення надійного перемикання в схемах деяких тригерів у точках А необхідно розміщувати елементи затримки. Для затримки сигналів використовують бістабільну схему того самого типу, що вже є у тригері.

Можливі простіші варіанти тригерних схем, коли у структурі тригера відсутня, наприклад, СК і тригер не має тактових входів.

Далі розглядатимуться тригери, що мають один тактовий вхід (синхровхід). Такі тригери називають однотактними.

Бістабільна схема

Основною структурною одиницею тригера є бістабільна схема. Є два різновиди бістабільної схеми – бістабільна схема на елементах І-НЕ (рис. 2.2) та бістабільна схема на елементах АБО-НЕ (рис. 2.3).

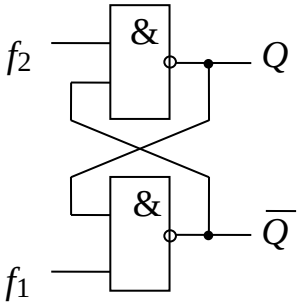


Рис. 2.2. Бістабільна схема на елементах І-НЕ

Таблиця 2.1. Таблиця функцій збудження бістабільної схеми на елементах І-НЕ

$Q(t_i)$	$Q(t_{i+1})$	f_2	f_1
0	0	1	×
0	1	0	1
1	0	1	0
1	1	×	1

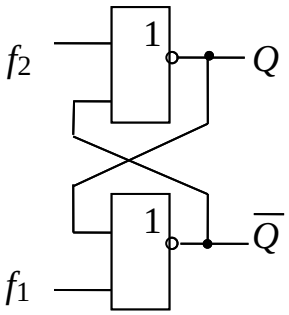


Рис. 2.3. Бістабільна схема на елементах АБО-НЕ

Таблиця 2.2. Таблиця функцій збудження бістабільної схеми на елементах АБО-НЕ

$Q(t_i)$	$Q(t_{i+1})$	f_2	f_1
0	0	×	0
0	1	0	1
1	0	1	0
1	1	0	×

Якщо бістабільна схема на елементах І–НЕ у момент часу t_i перебуває у стані ′0′ (на виході Q – 0, на виході $\overline{Q}$ – 1), а у наступний (t_{i+1}) момент вона має зберегти свій стан ($Q(t_{i+1}) = Q(t_i) = 0$), то на вхід f_2 необхідно

подати 1 (табл. 2.1), а на f_1 – будь–яке значення (×). Для переходу схеми з ′0′ в ′1′ ($Q(t_{i+1}) = 1$) на вхід f_2 необхідно подати 0, а на f_1 – 1. Якщо схема перебуває в ′1′ ($Q(t_i) = 1$) , то для її переходу у стан ′0′ ($Q(t_{i+1}) = 0$) на вхід f_2 має бути подано сигнал 1, на вхід f_1 – сигнал 0; для збереження стану ′1′ необхідно, щоб $f_2 = \times$, $f_1 = 1$.

Аналогічним чином можна дослідити роботу бістабільної схеми на елементах АБО-НЕ (табл. 2.3).

Обидві бістабільні схеми використовуються в тригерах, що випускаються у вигляді мікросхем.

Тригери найчастіше класифікують за функціональним призначенням і за способом запису інформації.

Класифікація тригерів за функціональним призначенням

Функціональне призначення тригера характеризують таблицею переходів, яка реалізує функцію $Q(t_{i+1}) = \lambda(Q(t_i), x(t_i))$, де λ – функція переходів тригера, $Q(t_i)$ – значення вихідного сигналу в момент часу t_i, $Q(t_{i+1})$ – значення вихідного сигналу в наступний момент часу; $x(t_i)$ – значення вхідних сигналів тригера в момент часу t_i.

З функціональної точки зору розрізняють RS-, R-, S-, E-, D-, T-, JK-, DV- тригери (таблиці переходів 2.3 – 2.10) та інші.

Аналізуючи таблицю переходів, наприклад, RS-тригера, слід зробити висновок, що тригер не змінює свого стану в момент часу t_{i+1} ($Q(t_{i+1}) = Q(t_i)$), якщо в момент часу t_i на входи S та R подати сигнали $S(t_i) = 0$, $R(t_i) = 0$. При дії сигналів $S(t_i) = 0$, $R(t_i) = 1$ тригер у момент часу t_{i+1} перейде у нульовий стан ($Q(t_{i+1}) = 0$), а при $S(t_i) = 1$, $R(t_i) = 0$ – у стан ′1′ ($Q(t_{i+1}) = 1$). При $S(t_i) = 1$, $R(t_i) = 1$ стан тригера є невизначеним ($Q(t_{i+1}) = -$). Така комбінація сигналів для RS- тригера є забороненою.

S-, R-, E- тригери відрізняються від RS-тригера тим, що при $S(t_i) = R(t_i) = 1$ S-тригер у наступний (t_{i+1}) момент часу переходить у стан ′1′ (табл. 2.5), R-тригер – у нульовий стан (табл. 2.4), а E-тригер не змінює свого стану (табл. 2.6).

D-тригер (табл. 2.10) називають тригером затримки, для нього справедлива рівність $Q(t_{i+1}) = D(t_i)$.

DV-тригер при $V = 1$ працює як D-тригер (табл. 2.8), а при $V = 0$ не змінює свого стану.

T-тригер (табл. 2.9) при $T = 0$ не змінює свого стану, а при $T = 1$ у наступний момент часу переходить у протилежний стан ($Q(t_{i+1}) = \overline{Q}(t_i)$). Та-

кий тригер виконує додавання вхідних сигналів за модулем 2 і його часто називають лічильним тригером.

JK-тригер (табл. 2.7) при комбінаціях вхідних сигналів $J = K = 0$; $J = 0$, $K = 1$; $J = 1$, $K = 0$ працює як RS-тригер (вхід J тотожний входові S, а K – входові R), а при $J = K = 1$ у наступний (t_{i+1}) момент часу змінює свій стан на протилежний і, отже, працює як лічильний тригер.

Таблиця 2.3. Таблиця переходів RS-тригера

$S(t_i)$	$R(t_i)$	$Q(t_{i+1})$
0	0	$Q(t_i)$
0	1	0
1	0	1
1	1	–

Таблиця 2.4. Таблиця переходів R-тригера

$S(t_i)$	$R(t_i)$	$Q(t_{i+1})$
0	0	$Q(t_i)$
0	1	0
1	0	1
1	1	0

Таблиця 2.5. Таблиця переходів S-тригера

$S(t_i)$	$R(t_i)$	$Q(t_{i+1})$
0	0	$Q(t_i)$
0	1	0
1	0	1
1	1	1

Таблиця 2.6. Таблиця переходів E-тригера

$S(t_i)$	$R(t_i)$	$Q(t_{i+1})$
0	0	$Q(t_i)$
0	1	0
1	0	1
1	1	$Q(t_i)$

Таблиця 2.7. Таблиця переходів JK-тригера

$J(t_i)$	$K(t_i)$	$Q(t_{i+1})$
0	0	$Q(t_i)$
0	1	0
1	0	1
1	1	$\overline{Q}(t_i)$

Таблиця 2.8. Таблиця переходів DV-тригера

$D(t_i)$	$V(t_i)$	$Q(t_{i+1})$
0	0	$Q(t_i)$
1	0	$Q(t_i)$
0	1	0
1	1	1

Таблиця 2.9. Таблиця переходів T-тригера

$T(t_i)$	$Q(t_{i+1})$
0	$Q(t_i)$
1	$\overline{Q}(t_i)$

Таблиця 2.10. Таблиця переходів D-тригера

$D(t_i)$	$Q(t_{i+1})$
0	0
1	1

Кількість тригерів надзвичайно велика. Наприклад, якщо тригер має один інформаційний вхід (можливі стани на вході: $x = 0$ та $x = 1$; на виході: 0, 1, Q, $\overline{Q}$ та один невизначений стан), то можна одержати $5^2 = 25$ типів тригерів. У загальному випадку при n інформаційних входах можна одержати 5^{2^n} типів тригерних схем.

Класифікація тригерів за способом запису інформації

Класифікація тригерів за способом запису інформації характеризує хід процесу перемикання тригера (рис. 2.4).

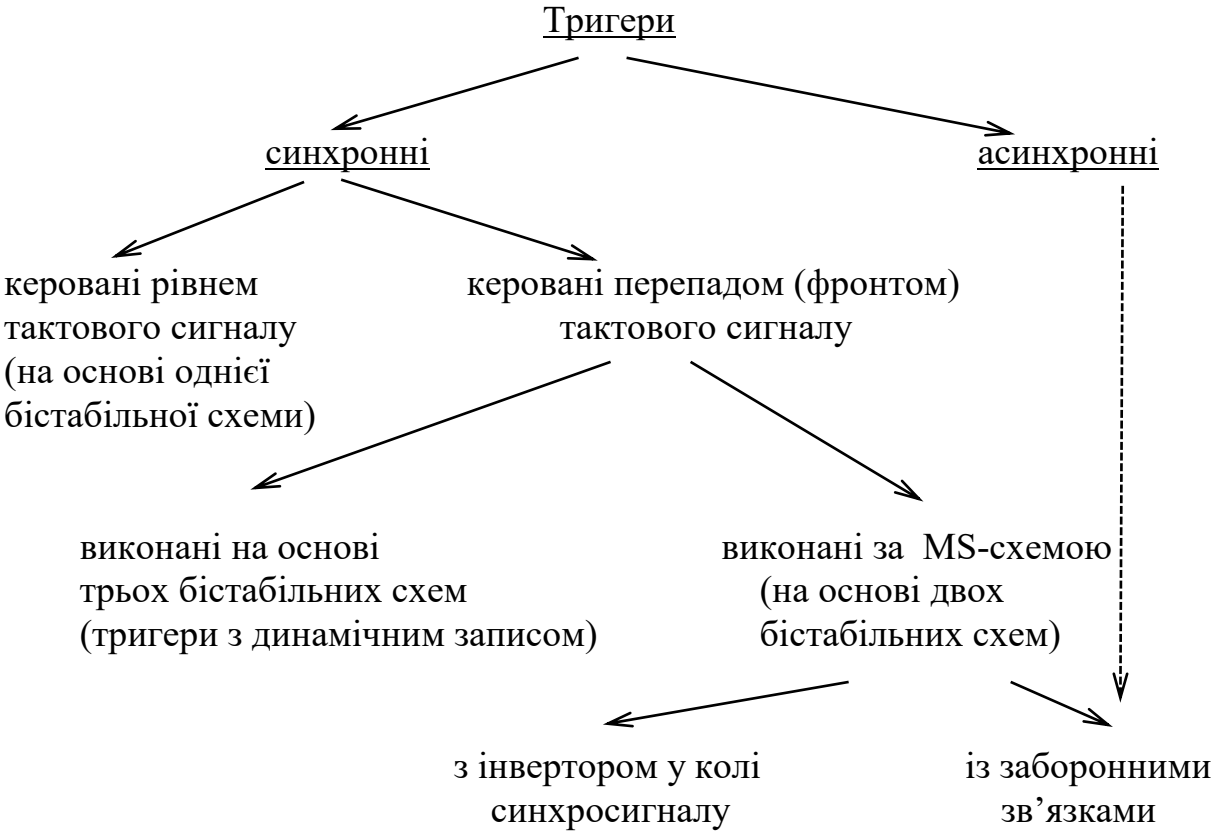


Рис. 2.4. Класифікація тригерів за способом запису інформації

Перш за все тригери поділяють на синхронні та асинхронні.

Асинхронні тригери не мають тактового входу, тому запис інформації в такі тригери здійснюється безпосередньо надходженням інформаційних сигналів.

Синхронні тригери мають тактові входи.

Розрізняють синхронні тригери, керовані рівнем тактового сигналу та перепадом (фронтом) тактового сигналу.

Тригери, керовані перепадом (фронтом) тактового сигналу, називають тригерами з внутрішньою затримкою.

Синхронні тригери, керовані рівнем тактового сигналу (рис. 2.5, 2.6), перемикаються відповідно до таблиці переходів тригера, якщо на синхровході має місце активний рівень тактового сигналу. До складу таких

тригерів входить лише одна бістабільна схема. Недоліком тригерів, керованих рівнем тактового сигналу, є те, що протягом дії активного рівня тактового сигналу тригер може перемикатися стільки разів, скільки разів при цьому змінюються сигнали на інформаційних входах тригера.

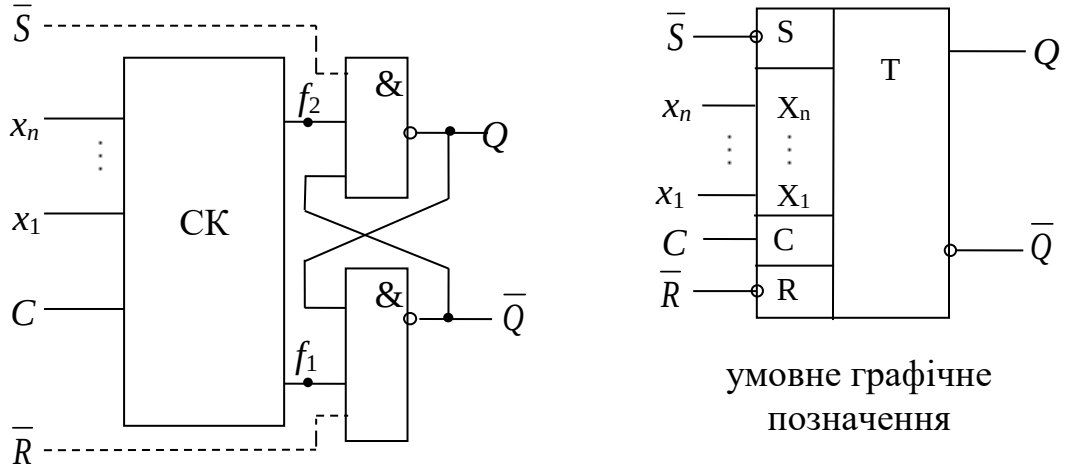


Рис. 2.5. Узагальнена структура тригера, керованого рівнем тактового сигналу, на елементах І-НЕ

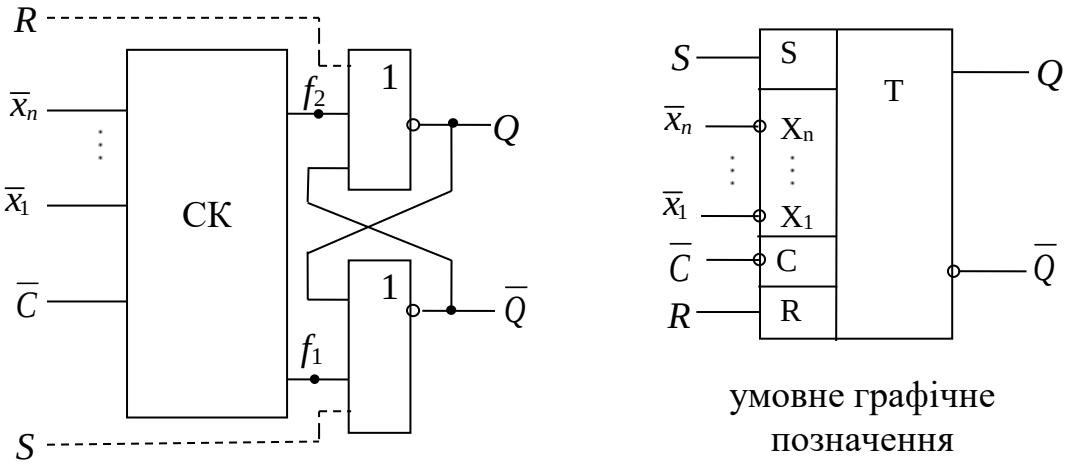


Рис. 2.6. Узагальнена структура тригера, керованого рівнем тактового сигналу, на елементах АБО-НЕ

На схемах показані входи асинхронного встановлення тригера в ′0′ та в ′1′ (входи $\overline{S}$, $\overline{R}$ – на рис. 2.5, входи S, R – на рис. 2.6). Входи асинхронного встановлення тригера в ′0′ та ′1′ використовуються перед початком роботи для того, щоб встановити тригер у потрібний початковий стан – ′0′ або ′1′, і з цього стану розпочати роботу (табл. 2.11, табл. 2.12).

У режимі «Робота» тригер функціонує за його таблицею переходів. Для переходу в цей режим на входах асинхронного встановлення тригера в ′0′, ′1′ має бути встановлена відповідна комбінація сигналів: $\overline{S}$ = 1, $\overline{R}$ = 1 – для тригерів на елементах І-НЕ (табл. 2.11); S = 0, R = 0 – для тригерів на елементах АБО-НЕ (табл. 2.12.). Далі тригер змінюватиме свій стан залежно від значень сигналів на його інформаційних входах (за таблицею переходів тригера).

Для будь-яких структур тригерів справедливе наступне ***правило заміни*** **І-НЕ** ***на*** **АБО-НЕ** (стосується лише елементів, а не змінних).

У схемі тригера елементи **І-НЕ** ***можна замінити на елементи*** **АБО-НЕ** ***і навпаки. При цьому в схемі тригера на елементах*** **АБО-НЕ** ***порівняно зі схемою тригера на елементах*** **І-НЕ** ***слід:***

- ***замість сигналу С подавати*** $\overline{C}$;
- ***входи*** $\overline{S}$, $\overline{R}$ ***асинхронного встановлення тригера в ’0’, ’1’ замінити на входи асинхронного встановлення R, S відповідно (*** $\overline{S}$ ***замінити на R;*** $\overline{R}$ ***замінити на S);***
- ***замість прямих змінних на інформаційних входах тригера використовувати інверсії змінних (див., наприклад, рис. 2.6), але у виразах функцій*** f_2, f_1 x_i ***замінювати на*** $\overline{x_i}$ ***у загальному випадку не можна.***

Таблиця 2.11. Режими роботи тригера на елементах І-НЕ

Входи асинхронного встановлення в ’0’, ’1’		Режим
$\overline{S}$	$\overline{R}$	
0	1	«Встановлення в ’1’» } перед початком роботи
1	0	«Встановлення в ’0’» } перед початком роботи
1	1	«Робота»
0	0	Заборонено

Примітка. Активним рівнем сигналів $\overline{S}$, $\overline{R}$ є рівень логічного нуля.

Таблиця 2.12. Режими роботи тригера на елементах АБО-НЕ

Входи асинхронного встановлення в ’0’, ’1’		Режим
S	*R*	
1	0	«Встановлення в ’1’» } перед початком роботи
0	1	«Встановлення в ’0’» } перед початком роботи
0	0	«Робота»
1	1	Заборонено

Примітка. Активним рівнем сигналів *S*, *R* є рівень логічної одиниці.

Тригери, керовані перепадом тактового сигналу (тригери з внутрішньою затримкою) містять у своєму складі дві або три бістабільні схеми. Вони позбавлені недоліку, притаманного тригерам, керованих рівнем тактового сигналу, оскільки переходять у новий стан лише у момент перепаду (переходу з 0 в 1 або з 1 в 0) тактового сигналу. Якщо тригер змінює свій стан при переході тактового сигналу з 0 в 1, то вважають що тригер спрацьовує за переднім фронтом тактового сигналу, а якщо при переході тактового сигналу з 1 в 0 – за заднім фронтом.

У схемах тригерів, керованих перепадом (фронтом) синхросигналу зміна інформаційних сигналів при встановленому рівні тактового сигналу (наприклад, стала ′1′ або сталий ′0′ тактового сигналу) не може спричинити перехід тригера у новий стан. При цьому, звичайно, можуть викликатися перемикання деяких логічних елементів схеми, однак на стан тригера це не впливає.

Потреба в тригерах з внутрішньою затримкою викликана тим, що під час проектування тригерних схем таблиці переходів проектованих тригерів можуть у стовпці $Q(t_{i+1})$ містити значення $\overline{Q}(t_i)$ (див., наприклад, табл. 2.7, 2.9). У цьому випадку сигнали на виходах Q, $\overline{Q}$ тригера (бістабільної схеми) є аргументами функцій збудження f_2, f_1 бістабільної схеми і для забезпечення правильного перемикання тригерів у точках А (рис. 2.1) необхідно розмістити елементи затримки (наприклад, додаткову бістабільну схему). Потреба в тригерах з внутрішньою затримкою викликана також тим, що під час проектування, наприклад, лічильників або регістрів зсуву, які звичайно складаються з тригерів, аргументами функцій f_2, f_1, у даному розряді лічильника або регістра є сигнали на

виходах Q та $\overline{Q}$ тригерів в інших розрядах, що мають перемикатися у процесі роботи одночасно з даним тригером.

Розрізняють два основні способи побудови тригерів, керованих фронтом синхросигналу:

- за MS-схемою (на основі двох бістабільних схем);
- на основі трьох бістабільних схем.

Можливі два варіанти побудови тригерів за* MS-*схемою: з інвертором у колі синхросигналу (рис. 2.7) та із заборонними зв′язками (рис. 2.8).

До складу тригера входять дві бістабільні схеми: М-БС (М – master) та S-БС (S – slave). Виходами тригера в цілому є виходи S-БС.

В обох випадках запис інформації в М-БС тактується сигналом C , а передача інформації з М-БС в S-БС здійснюється через вентилі **В**.

Якщо тригер будують на елементах І-НЕ, то вентилем є елемент І-НЕ, якщо тригер будують на елементах АБО-НЕ, то вентилем є елемент АБО-НЕ. У схемі тригера на елементах І-НЕ з інвертором у колі синхросигналу передача інформації (біта) з М-БС в S-БС відбувається в момент переходу тактового сигналу C з 1 в 0. Вентилі **В** при цьому відкриваються і стан М-БС переписується в S-БС. При $C = 0$ (сталий нуль) зміна інформаційних сигналів на входах x_n, …, x_1 не може вплинути на стан М-БС. При $C = 1$ (стала одиниця) стан М-БС під впливом інформаційних сигналів може змінитися, але це не позначиться на S-БС, оскільки **В** будуть закриті. І лише при наступному переході C з 1 в 0 новий стан М-БС буде переписано в S-БС і на виходах Q, $\overline{Q}$ встановляться нові значення.

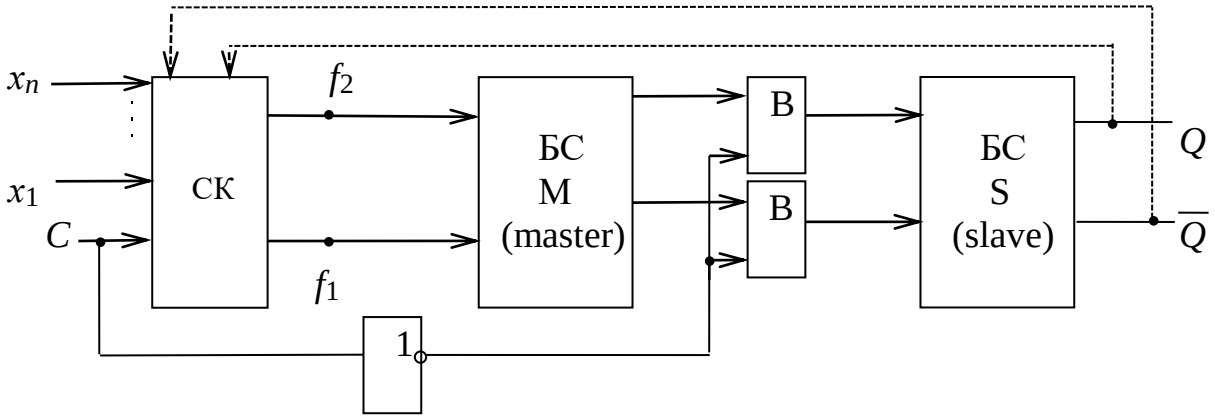


Рис. 2.7. Структура однотактного синхронного тригера з внутрішньою затримкою за MS-схемою з інвертором у колі синхросигналу

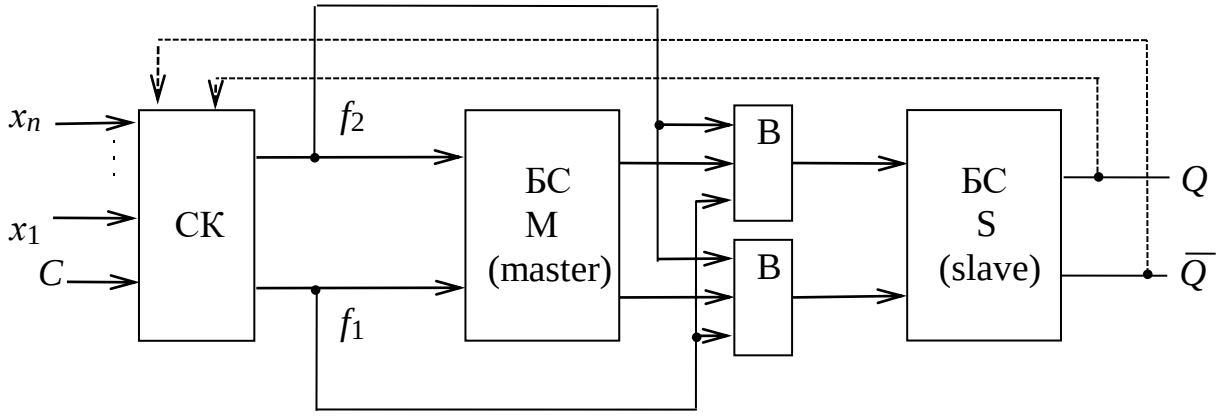


Рис. 2.8. Структура однотактного синхронного тригера з внутрішньою затримкою за MS-схемою із заборонними зв’язками

У схемі із заборонними зв′язками вентилі **В** відкриваються лише тоді, коли $f_2 = f_1$. Якщо $f_2 \neq f_1$, то обидва вентилі будуть закритими.

Рівність $f_2 = f_1$ виконується лише в момент переходу тактового сигналу C з 0 в 1, або з 1 в 0. У тригерах на елементах АБО-НЕ передача біта з М-БС у S-БС має місце, коли $f_2 = f_1 = 0$ (це досягається за переднім фронтом сигналу C). У тригерах на елементах І-НЕ передача інформації (біта) з М-БС у S-БС має місце, коли $f_2 = f_1 = 1$ (це досягається за заднім фронтом сигналу C).

При сталому значенні сигналу на вході C (сталий нуль або стала одиниця) М-БС та S-БС ізольовані одна від одної, що забезпечує стабільність роботи тригера.

За MS-схемою із заборонними зв′язками можуть проектуватися також асинхронні тригери, при цьому сигнал C має бути відсутнім.

Узагальнені структури тригерів за MS-схемою з інвертором у колі C та із заборонними зв’язками на елементах І-НЕ показані на рис. 2.9, 2.10. У цих схемах елементи І-НЕ можна замінити на елементи АБО-НЕ за сформульованим вище правилом заміни.

Іншим способом побудови синхронних тригерів, керованих перепадом (фронтом) тактового сигналу, є ***структура тригера на основі трьох бістабільних схем*** (рис. 2.11). Такий тригер часто називають тригером з динамічним записом.

Тригерний пристрій на елементах І-НЕ, побудований на основі трьох бістабільних схем (рис. 2.12), встановлюється у новий стан при переході тактового сигналу з 0 в 1 (за переднім фронтом).

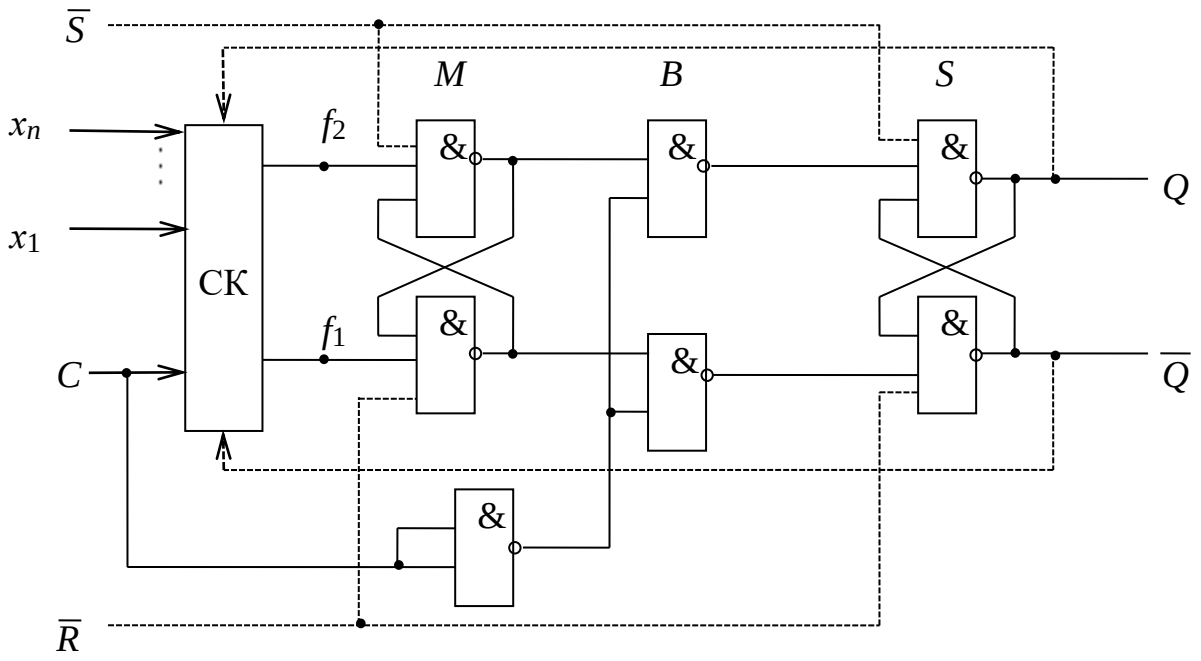


Рис. 2.9. Будова синхронного тригера за MS-схемою з інвертором у колі синхросигналу на елементах І-НЕ

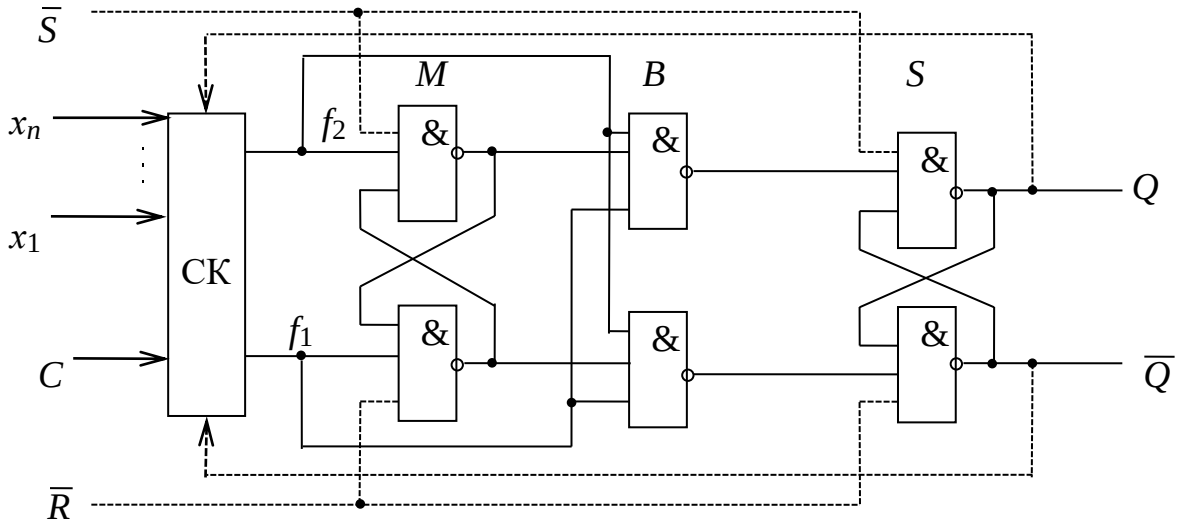


Рис. 2.10. Будова синхронного тригера за MS-схемою із заборонними зв′язками на елементах І-НЕ

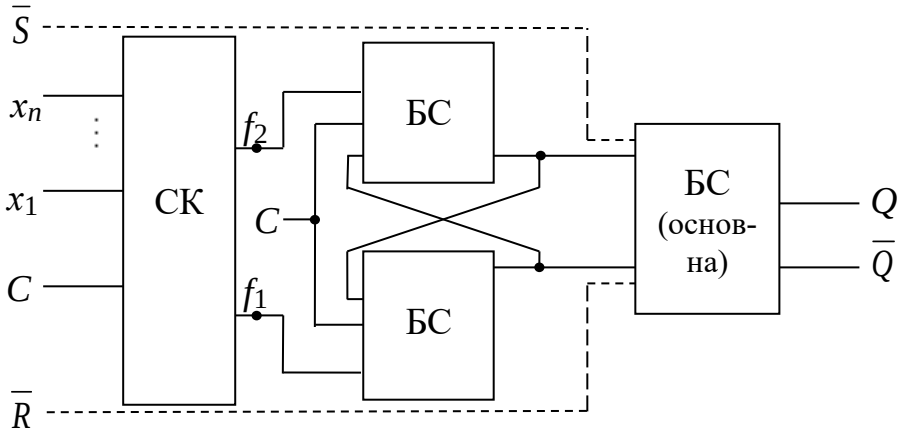


Рис. 2.11. Узагальнена структура тригера на основі трьох бістабільних схем (тригера з динамічним записом)

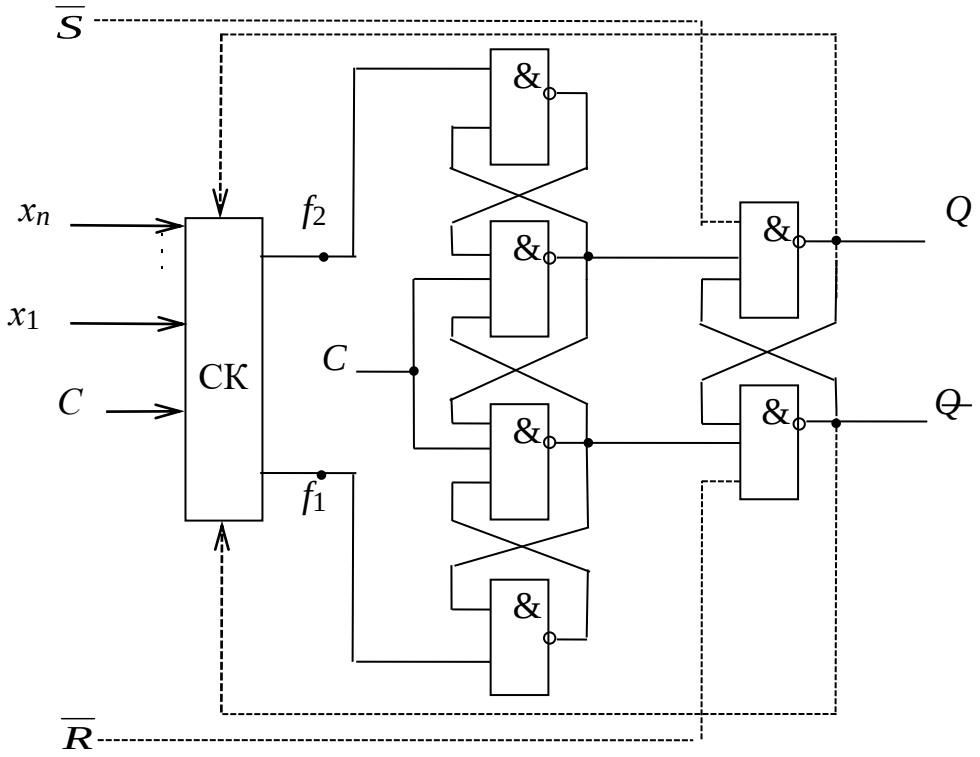


Рис. 2.12. Будова тригера на основі трьох бістабільних схем на елементах І-НЕ

У схемі на рис. 2.12 елементи І-НЕ можна замінити на елементи АБО-НЕ. При цьому входи $\overline{S}$, $\overline{R}$ асинхронного встановлення тригера в ’0’, ’1’ слід замінити на входи *R*, *S* відповідно, а замість сигналу *C* подавати $\overline{C}$.

Методика проектування тригерів

При проектуванні тригерів задають:

- тип тригера, який необхідно спроектувати (RS-, T-, D-, JK- тощо);
- тип логічних елементів, на основі яких має бути побудований тригер (І-НЕ або АБО-НЕ);
- спосіб запису інформації в тригер.

Проектування зводиться до:

1) вибору необхідної структури тригера:
   - тригер, керований рівнем тактового сигналу;
   - тригер на основі MS-схеми (один з двох варіантів – з інвертором у колі синхросигналу або із заборонними зв'язками);
   - тригер на основі трьох бістабільних схем (тригер з динамічним записом інформації);
2) синтезу схеми керування тригера – комбінаційної схеми з двома виходами, яка реалізує функції збудження f_2, f_1 тригера.

Методику проектування тригерів розглянемо на прикладі розв'язування задач.

Задача 1. Побудувати RS-тригер, керований рівнем тактового сигналу, на елементах АБО-НЕ, а також на елементах І-НЕ. На схемі тригера показати пунктиром асинхронні входи встановлення тригера в '0' та '1', а також навести умовне графічне позначення тригера.

Розв'язування. 1. Вибираємо структуру тригера, керованого рівнем тактового сигналу на елементах АБО-НЕ (рис. 2.6).

Далі синтезуємо схему керування тригера (комбінаційну схему з двома виходами).

2. Беремо таблицю переходів RS-тригера (табл. 2.3) і таблицю функцій збудження бістабільної схеми на елементах АБО-НЕ (табл. 2.2):

таблиця переходів RS-тригера

$S(t_i)$	$R(t_i)$	$Q(t_{i+1})$
0	0	$Q(t_i)$
0	1	0
1	0	1
1	1	–

таблиця функцій збудження бістабільної схеми на елементах АБО-НЕ

$Q(t_i)$	$Q(t_{i+1})$	f_2	f_1
0	0	×	0
0	1	0	1
1	0	1	0
1	1	0	×

3. Будуємо повну таблицю переходів синхронного RS-тригера (табл. 2.13).

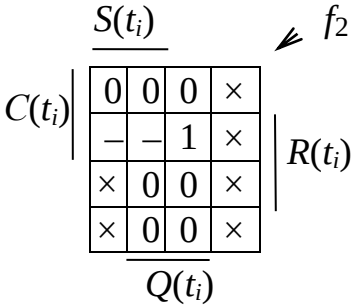
При $C = 0$ тригер не змінює свого стану, тому $Q(t_{i+1}) = Q(t_i)$. При $C = 1$ тригер має функціонувати за таблицею переходів RS-тригера. Аналізуючи значення $S(t_i)$, $R(t_i)$ у кожному рядку нижньої частини табл. 2.13 заповнюємо стовпець $Q(t_{i+1})$, керуючись таблицею переходів RS-тригера.

4. У повній таблиці переходів проектованого тригера, аналізуючи порядково переходи $Q(t_i) \to Q(t_{i+1})$ і беручи до уваги таблицю функцій збудження бістабільної схеми на елементах АБО-НЕ, заповнюємо стовпці f_2 та f_1.

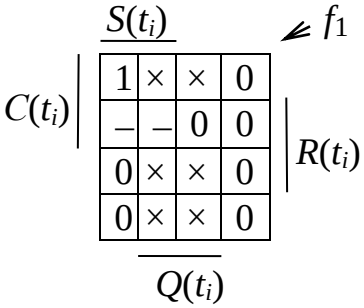
Таблиця 2.13. Повна таблиця переходів синхронного RS-тригера на елементах АБО-НЕ

$C(t_i)$	$S(t_i)$	$R(t_i)$	$Q(t_i)$	$Q(t_{i+1})$	f_2	f_1
0	0	0	0	0	×	0
0	0	0	1	1	0	×
0	0	1	0	0	×	0
0	0	1	1	1	0	×
0	1	0	0	0	×	0
0	1	0	1	1	0	×
0	1	1	0	0	×	0
0	1	1	1	1	0	×
1	0	0	0	0	×	0
1	0	0	1	1	0	×
1	0	1	0	0	×	0
1	0	1	1	0	1	0
1	1	0	0	1	0	1
1	1	0	1	1	0	×
1	1	1	0	–	–	–
1	1	1	1	–	–	–

5. За допомогою діаграм Вейча мінімізуємо функції f_2 та f_1:



$f_2 = R \cdot C$, $f_2 = \overline{\overline{R} \vee \overline{C}}$.



$f_1 = S \cdot C$, $f_1 = \overline{\overline{S} \vee \overline{C}}$.

6. Будуємо схему керування тригера – комбінаційну схему, яка реалізує функції f_2, f_1 (рис. 2.13).

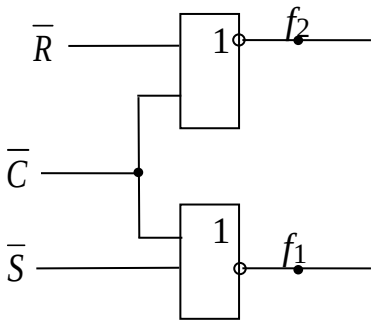
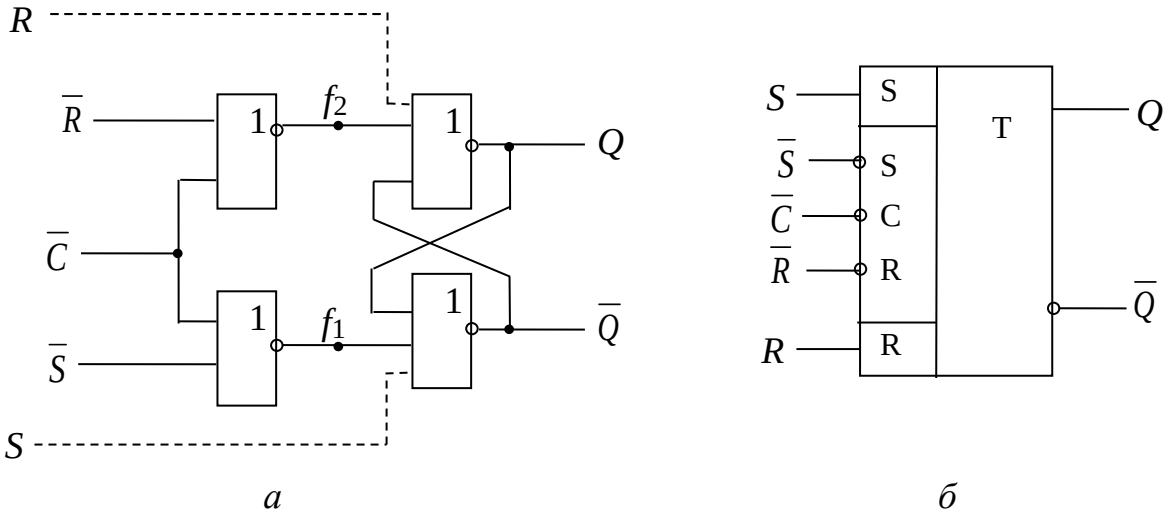


Рис. 2.13. Схема керування RS-тригера на елементах АБО-НЕ

7. Будуємо схему тригера (рис. 2.14). Для цього в узагальненій структурі тригера на рис. 2.6. замість СК вставляємо щойно спроектовану комбінаційну схему з двома виходами. У підсумку отримаємо функціональну схему тригера на рис. 2.14.



а *б*

Рис. 2.14. Синхронний RS-тригер на елементах АБО-НЕ, керований рівнем тактового сигналу (входи *S*, *R* асинхронного встановлення тригера в ’0’, ’1’ можуть бути відсутні):
а – функціональна схема тригера; *б* – умовне графічне позначення тригера

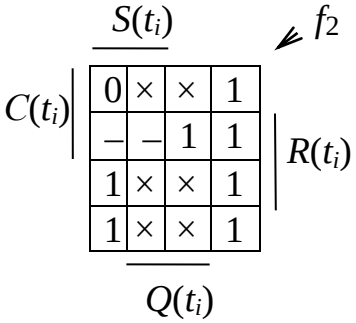
При побудові тригера на елементах І-НЕ беремо таблицю функцій збудження бістабільної схеми на елементах І-НЕ (табл. 2.1) та будуємо повну таблицю переходів тригера (табл. 2.14) (у верхній частині таблиці (ко-

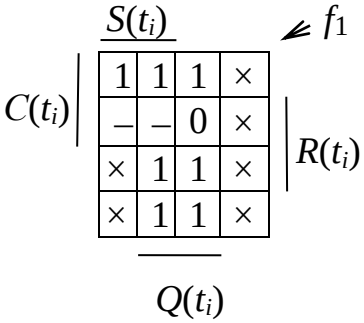
ли $C = 0$) $Q(t_{i+1}) = Q(t_i)$; у нижній (коли $C = 1$) – стовпець $Q(t_{i+1})$ заповнюємо керуючись таблицею переходів RS-тригера).

Таблиця 2.14. Повна таблиця переходів синхронного RS-тригера на елементах І-НЕ

$C(t_i)$	$S(t_i)$	$R(t_i)$	$Q(t_i)$	$Q(t_{i+1})$	f_2	f_1
0	0	0	0	0	1	×
0	0	0	1	1	×	1
0	0	1	0	0	1	×
0	0	1	1	1	×	1
0	1	0	0	0	1	×
0	1	0	1	1	×	1
0	1	1	0	0	1	×
0	1	1	1	1	×	1
1	0	0	0	0	1	×
1	0	0	1	1	×	1
1	0	1	0	0	1	×
1	0	1	1	0	1	0
1	1	0	0	1	0	1
1	1	0	1	1	×	1
1	1	1	0	–	–	–
1	1	1	1	–	–	–

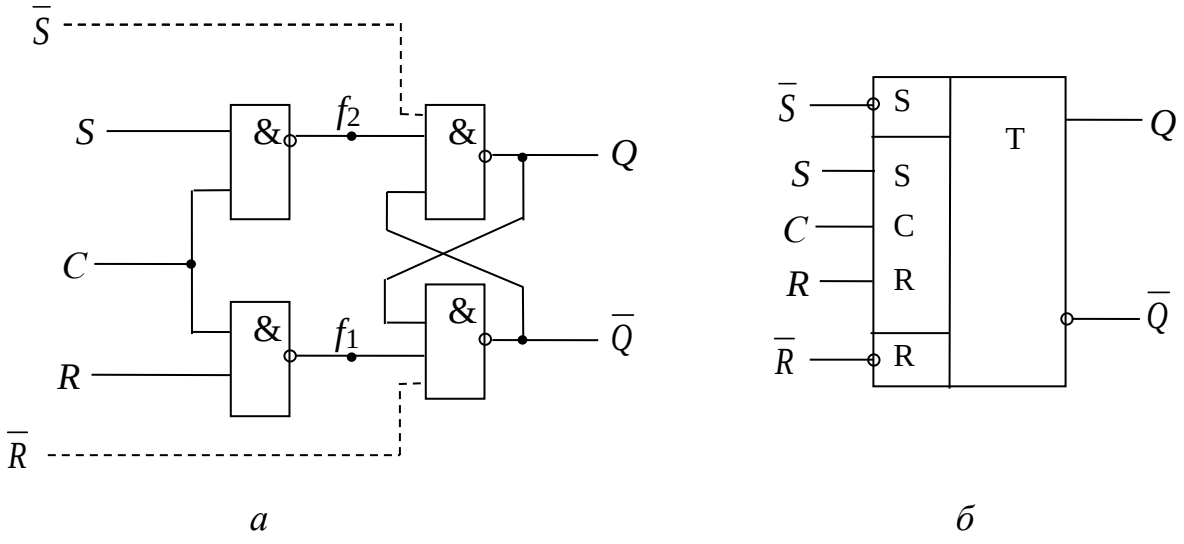
Заповнюємо стовпці f_2, f_1 табл. 2.14., беручи до уваги таблицю функцій збудження бістабільної схеми на елементах І-НЕ, та мінімізуємо функції f_2, f_1:





$$f_2 = \overline{S} \vee \overline{C} = \overline{SC} \qquad f_1 = \overline{R} \vee \overline{C} = \overline{RC}$$

Далі за отриманим рівнянням для f_2, f_1 та беручи за прототип рис. 2.5 будуємо схему тригера (рис. 2.15).



а *б*

Рис. 2.15. Синхронний RS-тригер на елементах І–НЕ, керований рівнем тактового сигналу (входи $\overline{S}$, $\overline{R}$ асинхронного встановлення тригера в ’0’, ’1’ можуть бути відсутні):
а – функціональна схема тригера; *б* – умовне графічне позначення тригера

Задача 2. Побудувати RS-тригер на елементах АБО-НЕ за MS-схемою із заборонними зв’язками. На схемі тригера показати пунктиром асинхронні входи встановлення тригера в ’0’ та ’1’, а також навести умовне графічне позначення тригера.

Розв’язування. 1. Вибираємо за прототип структуру тригера на рис. 2.10.

Далі синтезуємо схему керування тригера, яка реалізує функції збудження f_2, f_1.

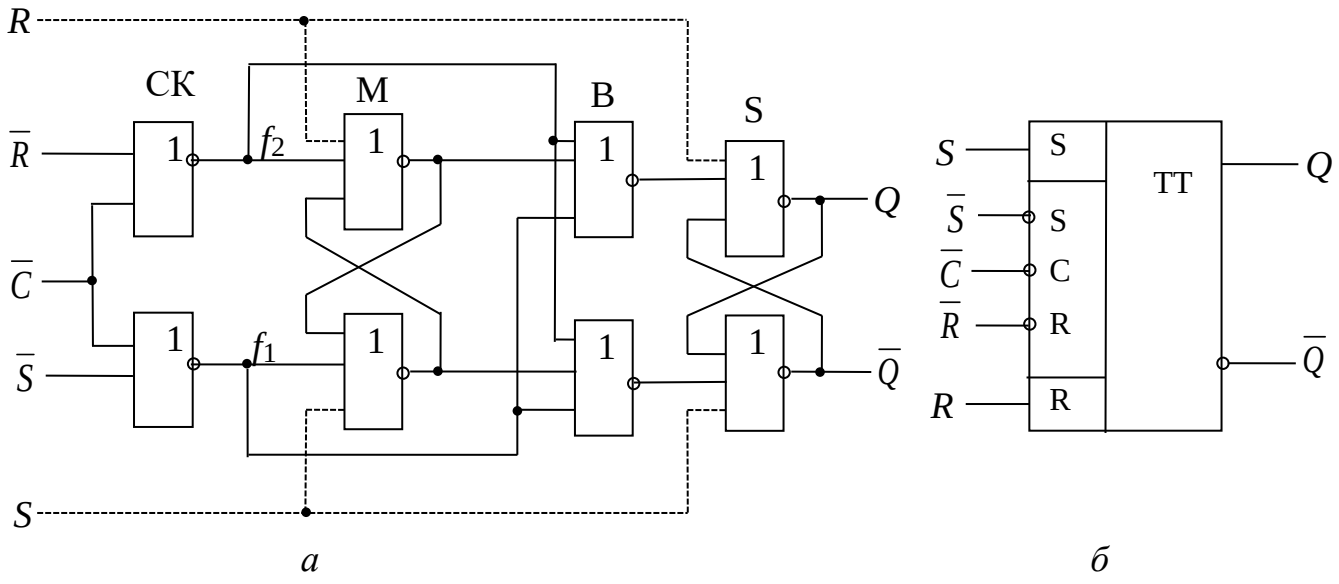
Синтез схеми керування повністю співпадає з пунктами 2–6 задачі 1.

7. Будуємо схему тригера (рис. 2.16). Для цього в узагальненій структурі на рис. 2.10 замінюємо елементи І-НЕ на АБО-НЕ. При цьому замість сигналу C подається $\overline{C}$, асинхронний вхід $\overline{S}$ слід назвати R, асинхронний вхід $\overline{R}$ слід назвати S, а замість СК вставляємо комбінаційну схему з рис. 2.13.

Перехід тригера з одного стану в інший відбувається за заднім фронтом синхросигналу.

Задача 3. Спроектувати синхронний Т-тригер на елементах І-НЕ на основі трьох бістабільних схем.

Розв’язування. 1. Вибираємо узагальнену структуру тригера на основі трьох бістабільних схем (рис.2.12).



а *б*

Рис. 2.16. RS-тригер на елементах АБО-НЕ, виконаний за MS-схемою із заборонними зв’язками:
а – функціональна схема тригера; *б* – умовне графічне позначення тригера

2. Беремо таблицю переходів Т-тригера (табл. 2.9) та таблицю функцій збудження бістабільної схеми на елементах І-НЕ (табл. 2.1):

таблиця переходів Т-тригера

$T(t_i)$	$Q(t_{i+1})$
0	$Q(t_i)$
1	$\overline{Q}(t_i)$

таблиця функцій збудження бістабільної схеми на елементах І-НЕ

$Q(t_i)$	$Q(t_{i+1})$	f_2	f_1
0	0	1	×
0	1	0	1
1	0	1	0
1	1	×	1

3. Будуємо повну таблицю переходів синхронного Т-тригера (табл. 2.15). При $C = 0$ тригер не змінює свого стану, тому $Q(t_{i+1}) = Q(t_i)$. При $C = 1$ тригер має функціонувати за таблицею переходів Т-тригера. Аналізуючи значення $T(t_i)$ у кожному рядку нижньої частини табл. 2.15 заповнюємо стовпець $Q(t_{i+1})$, керуючись таблицею переходів Т-тригера.

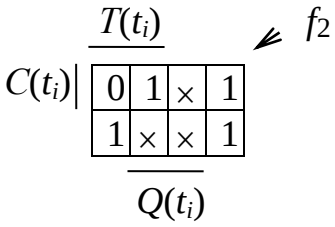
4. У повній таблиці переходів проектованого тригера, аналізуючи порядково переходи $Q(t_i) \to Q(t_{i+1})$ і беручи до уваги таблицю функцій

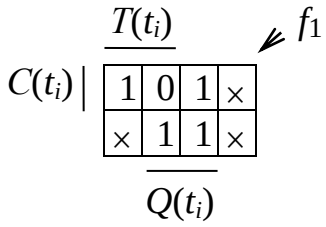
збудження бістабільної схеми на елементах І–НЕ, заповнюємо стовпці f_2 та f_1.

Таблиця 2.15. Повна таблиця переходів синхронного Т-тригера на елементах І-НЕ

$C(t_i)$	$T(t_i)$	$Q(t_i)$	$Q(t_{i+1})$	f_2	f_1
0	0	0	0	1	×
0	0	1	1	×	1
0	1	0	0	1	×
0	1	1	1	×	1
1	0	0	0	1	×
1	0	1	1	×	1
1	1	0	1	0	1
1	1	1	0	1	0

5. За допомогою діаграм Вейча мінімізуємо функції f_2 та f_1:





$f_2 = \overline{T} \vee \overline{C} \vee Q$, $f_2 = \overline{C\,T\,\overline{Q}}$. $f_1 = \overline{T} \vee \overline{C} \vee \overline{Q}$, $f_1 = \overline{C\,T\,Q}$.

6. Будуємо схему керування тригера (рис. 2.17).

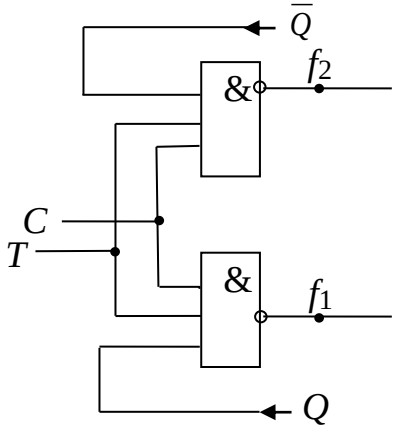


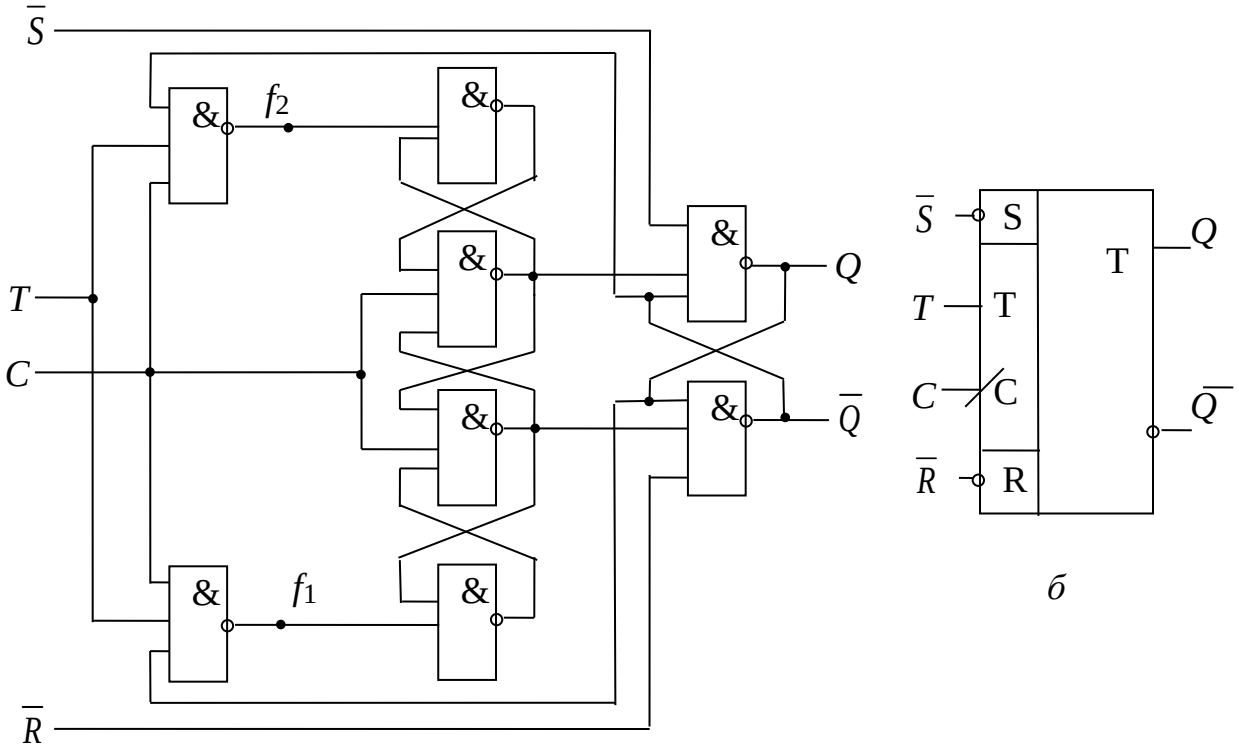
Рис. 2.17. Схема керування синхронного Т-тригера на елементах І-НЕ

7. Будуємо схему тригера (рис. 2.18). Для цього в узагальненій структурі на рис. 2.12 замість СК вставляємо щойно спроектовану комбінаційну схему з двома виходами (рис. 2.17).

Перехід тригера з одного стану в інший відбувається за переднім фронтом синхросигналу (коса лінія з нахилом вправо на вході *C* умовного графічного позначення тригера).

В умовних графічних позначеннях тригерів інформаційні входи показують в одному полі з тактовим входом *C*, а асинхронні входи встановлення тригера в ′0′ та ′1′ – в окремих полях (рис. 2.19). Якщо активним рівнем сигналу на деякому вході є рівень логічного нуля, то відповідний вхід позначається кружком. Наприклад, на рис. 2.19 встановлення RS- та JK-тригера в нуль (вхід $\overline{R}$) та в одиницю (вхід $\overline{S}$) здійснюється сигналом логічного нуля, аналогічно для Т-тригера активним рівнем сигналу на інформаційному вході є також рівень логічного нуля.

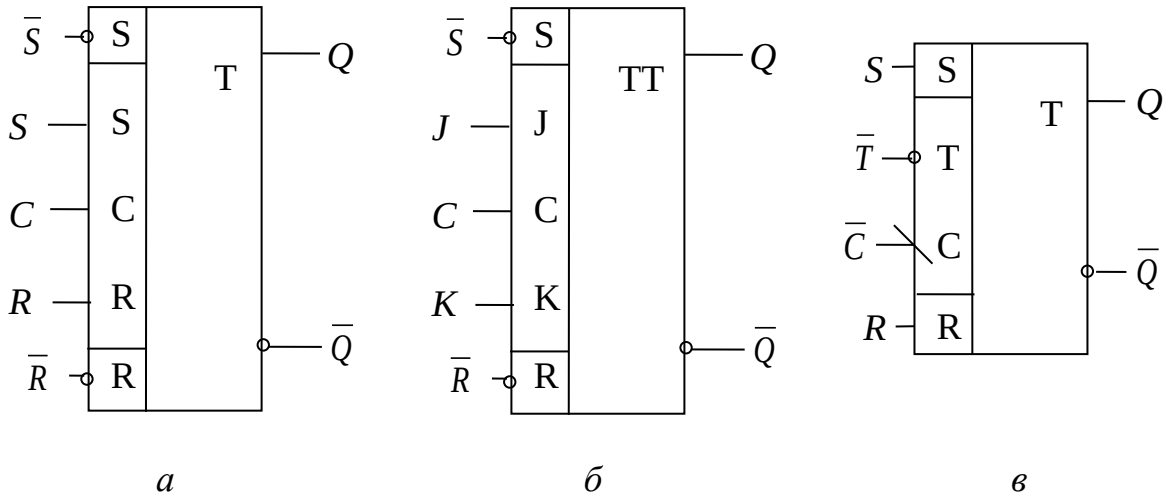
Якщо входи тригерів зображені без кружка (прямий вхід), то активним рівнем сигналу на таких входах є рівень логічної одиниці.



а *б*

Рис. 2.18. Синхронний Т-тригер на елементах І-НЕ, виконаний на основі трьох бістабільних схем:
а – функціональна схема тригера; *б* – умовне графічне позначення тригера

Тактовий вхід тригерів, побудованих на основі трьох бістабільних схем, додатково позначається косою лінією з нахилом вліво, якщо перемикання тригера здійснюється за заднім фронтом тактового сигналу (перехід C з 1 в 0), або з нахилом вправо, якщо перемикання тригера відбувається за переднім фронтом сигналу C (перехід C з 0 в 1). Таке позначення не використовують для тактового входу тригерів, виконаних за MS-схемою або керованих рівнем тактового сигналу. Крім того, тригери, виконані за MS-схемою, позначають літерами ТТ (рис. 2.19, б).



а *б* *в*

Рис. 2.19. Приклади умовного графічного позначення тригерів:
а – RS-тригер, керований рівнем тактового сигналу (на елементах І-НЕ);
б – JK-тригер, виконаний за MS-схемою (на елементах І-НЕ);
в – Т-тригер, виконаний на основі трьох бістабільних схем (на елементах АБО-НЕ)

Завдання на лабораторну роботу

1. Побудувати схеми тригерів, керованих рівнем тактового сигналу :
   S-, E- тригер – на елементах АБО-НЕ;
   RS-, R- тригер – на елементах І-НЕ.
2. Побудувати схеми тригерів з внутрішньою затримкою за MS-схемою:
   D-, JK- тригер з інвертором у колі синхросигналу на елементах І-НЕ;
   T-, DV- тригер із заборонними зв’язками на елементах АБО-НЕ.
3. На заданих логічних елементах побудувати тригер на основі трьох бістабільних схем (тригер з динамічним записом) з двома інформаційними входами x_2, x_1, який функціонує за заданою таблицею переходів (табл. 2.16).

Варіант – дві молодші цифри номера залікової книжки.

Якщо варіантом є парне число, то для побудови тригера використати елементи І-НЕ, якщо непарне – елементи АБО-НЕ.

Таблиця 2.16

Таблиця варіантів для побудови тригера на основі трьох бістабільних схем

$x_2(t_i)$	$x_1(t_i)$	$Q(t_{i+1})$																
		Варіант																
		1, 18	2, 19	3, 20	4, 21	5, 22	6, 23	7, 24	8, 25	9, 26	10, 27	11, 28	12, 29	13, 30	14, 31	15, 32	16, 33	17, 34
0	0	$Q(t_i)$	$\bar{Q}(t_i)$	$Q(t_i)$	0	0	1	$\bar{Q}(t_i)$	0	1	0	1	$\bar{Q}(t_i)$	$\bar{Q}(t_i)$	1	0	1	$\bar{Q}(t_i)$
0	1	$\bar{Q}(t_i)$	0	$\bar{Q}(t_i)$	1	1	0	1	1	$\bar{Q}(t_i)$	0	$Q(t_i)$	1	0	0	$Q(t_i)$	$\bar{Q}(t_i)$	$Q(t)$
1	0	0	1	1	$Q(t_i)$	$Q(t_i)$	0	0	$\bar{Q}(t_i)$	$Q(t_i)$	1	$\bar{Q}(t_i)$	0	0	$Q(t_i)$	$\bar{Q}(t_i)$	0	0
1	1	1	$Q(t_i)$	0	$\bar{Q}(t_i)$	$\bar{Q}(t_i)$	$\bar{Q}(t_i)$	0	0	0	$\bar{Q}(t_i)$	0	$Q(t_i)$	1	$\bar{Q}(t_i)$	1	0	1

4. Для кожної схеми тригера, користуючись табл. 1.4, розрахувати показник складності (кількість умовних корпусів) та швидкодії (час перемикання тригера з одного стану в інший).
5. Для всіх схем у п. 1 – п. 3 завдання пунктирними лініями показати асинхронні входи встановлення тригерів у ′0′ та ′1′, а також навести умовні графічні позначення тригерів.

Порядок виконання роботи

1. За завданням викладача побудувати модель одного з тригерів, керованих рівнем тактового сигналу (п. 1 завдання), перевірити його працездатність та дослідити часову діаграму роботи.
2. За завданням викладача побудувати модель одного з тригерів, побудованих за MS-схемою (п. 2 завдання), перевірити його працездатність та дослідити часову діаграму роботи.
3. Побудувати модель тригера на основі трьох бістабільних схем, заданого табл. 2.16 (п. 3 завдання), перевірити його працездатність та дослідити часову діаграму роботи.

Вимоги до оформлення звіту

Звіт має включати:

1) синтез чотирьох схем тригерів, керованих рівнем тактового сигналу (п. 1 завдання); чотирьох схем тригерів, побудованих за MS-схемою (п. 2

завдання); схему тригера на основі трьох бістабільних схем, заданого табл. 2.16 (п. 3 завдання), а також умовні графічні позначення тригерів;

2) розрахунок показників складності та швидкодії для дев'яти вищеперелічених схем тригерів;

3) часові діаграми роботи трьох моделей тригерів, досліджуваних в лабораторії, – керованого рівнем тактового сигналу, побудованого за MS-схемою, побудованого на основі трьох бістабільних схем.

Питання для самоперевірки

1. Нарисувати узагальнену структуру тригера.
2. За якими ознаками класифікують тригери?
3. Чим відрізняються синхронні тригери, керовані рівнем тактового сигналу, від тригерів, керованих перепадом (фронтом) тактового сигналу?
4. Записати узагальнену функцію переходів тригера.
5. Скласти таблиці переходів RS-, R-, S-, E-, D-, DV-, T- та JK-тригера.
6. Як класифікують тригери за способом запису інформації?
7. Нарисувати структуру синхронного тригера, виконаного за MS-схемою з інвертором у колі синхросигналу.
8. Нарисувати структуру синхронного тригера, виконаного за MS-схемою із заборонними зв′язками.
9. Побудувати таблицю функцій збудження бістабільної схеми на елементах І-НЕ.
10. Побудувати таблицю функцій збудження бістабільної схеми на елементах АБО-НЕ.
11. Нарисувати структуру синхронного тригера, виконаного на основі трьох бістабільних схем на елементах І-НЕ, і пояснити процес перемикання тригера з одного стану в інший.
12. Пояснити процес перемикання з одного стану в інший тригера на елементах АБО-НЕ, виконаного на основі трьох бістабільних схем.
13. Сформулювати методику проектування тригерів.
14. У чому полягає різниця між синхронними та асинхронними тригерами?
15. Пояснити різницю між інформаційними та асинхронними входами синхронного тригера.

Рекомендована література

1. Самофалов К.Г., Корнейчук В.И., Тарасенко В.П., Жабин В.И. Цифровые ЭВМ. Практикум. – К. : Вища школа, 1990. – 216 с.

2. Букреев И.Н., Мансуров В.М., Горячев В.И. Микроэлектронные схемы цифровых устройств. – М. : Сов. Радио, 1975. – 368 с.
3. Жабін В.І., Жуков І.А., Клименко І.А., Ткаченко В.В. Прикладна теорія цифрових автоматів. – К. : Вид-во Нац. авіац. ун-ту “НАУ-друк”, 2009. – 360 с.
4. Проектирование цифровых вычислительных машин: Учеб. пособие для студентов вузов / Под ред. С.А. Майорова. – М. : Высшая шк., 1972. – 344 с.

ЛАБОРАТОРНА РОБОТА №3. ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ РЕГІСТРІВ НА ПОТЕНЦІАЛЬНИХ ЕЛЕМЕНТАХ

Мета роботи: вивчити схемні різновиди регістрів, мікрооперації, виконувані на регістрах, а також опанувати методику проектування та дослідження регістрів на потенціальних елементах.

Теоретичні відомості

Регістром називають операційний вузол, призначений для запису, зберігання та видачі слів, а також для виконання мікрооперацій над ними.

Регістр складається із запам'ятовувальних елементів (тригерів).

Кількість розрядів (тригерів) регістра називають його довжиною. В узагальненій структурі регістра (рис. 3.1) використовуються такі позначення: КС – комбінаційна схема, x_n, ..., x_1 – інформаційні входи регістра, z_n, ..., z_1 – інформаційні виходи регістра, y_m, ..., y_1 – сигнали мікрооперацій; a_n, ..., a_1; b_n, ..., b_1 – входи функцій збудження тригерів, C – тактовий сигнал, n – довжина регістра.

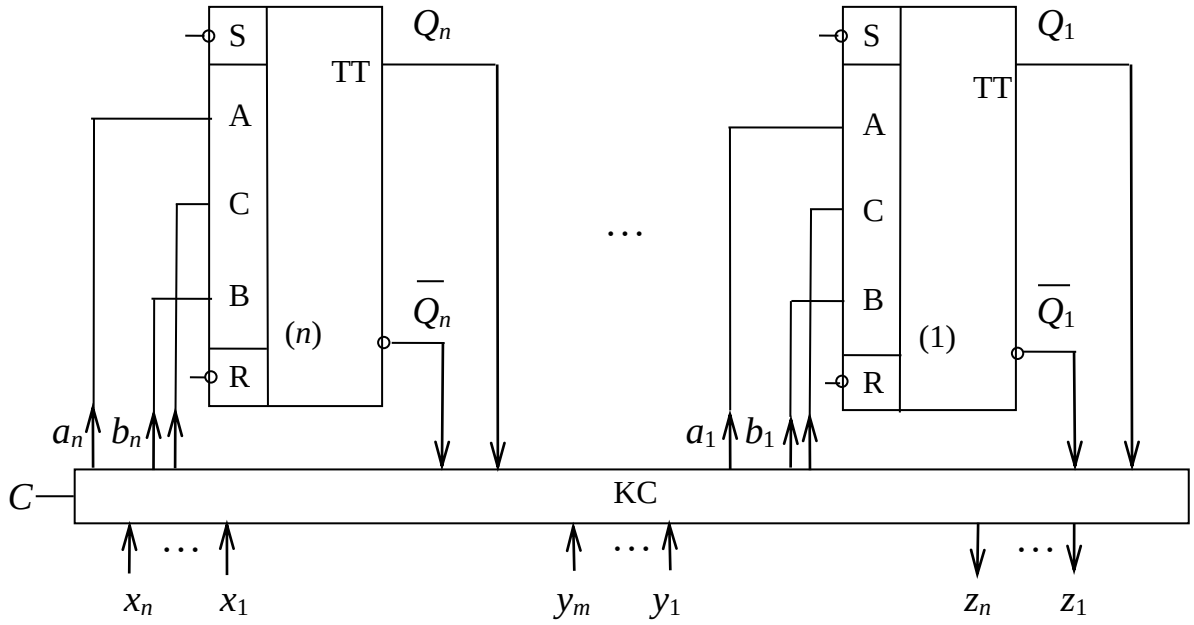


Рис. 3.1. Узагальнена структура регістра

Найчастіше на регістрах виконують такі мікрооперації:
RESET (y_1) – встановлення початкового стану '0…0';
SET (y_2) – встановлення початкового стану '1…1';
WRITE (y_3) – запис слова в регістр;

AND (y_4) – порозрядна кон'юнкція двох слів (одне слово знаходиться в регістрі, інше – надходить на інформаційні входи регістра);
OR (y_5) – порозрядна диз'юнкція двох слів;
XOR (y_6) – порозрядна сума двох слів за модулем два (eXclusive OR);
COM (y_7) – інвертування слова (COMpliment);
SLL (y_8) – логічний зсув слова на один розряд вліво (Shift Left Logical);
SRL (y_9) – логічний зсув слова на один розряд вправо (Shift Right Logical);
SLA (y_{10}) – арифметичний зсув на один розряд вліво (Shift Left Arithmetic);
SRA (y_{11}) – арифметичний зсув на один розряд вправо (Shift Right Arithmetic);
RL (y_{12}) – циклічний зсув слова на один розряд вліво (Rotation Left);
RR (y_{13}) – циклічний зсув слова на один розряд вправо (Rotation Right);
READ (y_{14}) – видача слова у прямому коді;
RDCOM (y_{15}) – видача слова в інверсному коді (ReaD COMpliment);
RDTRC (y_{16}) – видача слова у парафазному коді (ReaD Two Rail Code).

Регістри можна будувати на різноманітних тригерах: RS-, JK-, D-, T- тригерах тощо. Тригери можуть мати різну організацію – синхронні, асинхронні, з внутрішньою затримкою.

Складність КС регістра залежить від виду і кількості виконуваних мікрооперацій на регістрі та від типу тригера.

Регістри, на яких реалізована мікрооперація зсуву, називають регістрами зсуву.

Регістри, що забезпечують зсув слова як вліво, так і вправо, називають реверсивними регістрами.

Характеристика мікрооперацій

RESET (y_1) – встановити всі розряди регістра в ‘0’. Для виконання цієї мікрооперації необхідно на асинхронні входи $\overline{R}$ всіх тригерів одночасно подати логічний нуль.

SET (y_2) – встановити всі розряди регістра в ‘1’. Для виконання цієї мікрооперації необхідно на асинхронні входи $\overline{S}$ всіх тригерів одночасно подати логічний нуль.

WRITE (y_3) – запис слова в регістр :

$$Q_j(t_{i+1}) = x_j(t_i),$$

де $x_j(t_i)$ – значення сигналу на i-му вході регістра в момент часу t_i, $Q_j(t_{i+1})$ – стан j-го тригера (розряду) після виконання мікрооперації.

AND (y_4) – порозрядна кон'юнкція двох слів:

$$Q_j(t_{i+1}) = Q_j(t_i) \,\&\, x_j(t_i).$$

OR (y_5) – порозрядна диз'юнкція двох слів:

$$Q_j(t_{i+1}) = Q_j(t_i) \vee x_j(t_i).$$

XOR (y_6) – порозрядна сума двох слів за модулем два:

$$Q_j(t_{i+1}) = Q_j(t_i) \oplus x_j(t_i).$$

COM (y_7) – порозрядна інверсія слова:

$$Q_j(t_{i+1}) = \overline{Q}_j(t_i).$$

SLL (y_8) – логічний зсув слова на 1 розряд вліво:

$$\begin{cases} Q_j(t_{i+1}) = Q_{j-1}(t_i),\ j \neq 1 \\ Q_1(t_{i+1}) = \text{DR, DR (data right)} – \text{вхід заповнення молодшого розряду регістра.} \end{cases}$$

SRL (y_9) – логічний зсув слова на 1 розряд вправо:

$$\begin{cases} Q_j(t_{i+1}) = Q_{j+1}(t_i),\ j \neq n \\ Q_{n-1}(t_{i+1}) = \text{DL, DL (data left)} – \text{вхід заповнення старшого розряду регістра.} \end{cases}$$

SLA (y_{10}) – арифметичний зсув слова на 1 розряд вліво (число в доповняльному коді (ДК)):

$$\begin{cases} Q_j(t_{i+1}) = Q_{j-1}(t_i),\ j \neq n,\ j \neq 1 \\ Q_n(t_{i+1}) = Q_n(t_i) \\ Q_1(t_{i+1}) = 0. \end{cases}$$

SRA (y_{11}) – арифметичний зсув слова на 1 розряд вправо (число в ДК):

$$\begin{cases} Q_j(t_{i+1}) = Q_{j+1}(t_i),\ j \neq n \\ Q_n(t_{i+1}) = Q_n(t_i). \end{cases}$$

RL (y_{12}) – циклічний зсув слова на 1 розряд вліво:

$$\begin{cases} Q_j(t_{i+1}) = Q_{j-1}(t_i),\ j \neq 1 \\ Q_1(t_{i+1}) = Q_n(t_i). \end{cases}$$

RR (y_{13}) – циклічний зсув слова на 1 розряд вправо:

$$\begin{cases} Q_j(t_{i+1}) = Q_{j+1}(t_i), j \neq n \\ Q_n(t_{i+1}) = Q_1(t_i). \end{cases}$$

Мікрооперації видачі слова (стан регістра при цьому не змінюється):

READ (y_{14}) – видача слова у прямому коді (рис. 3.2, а):

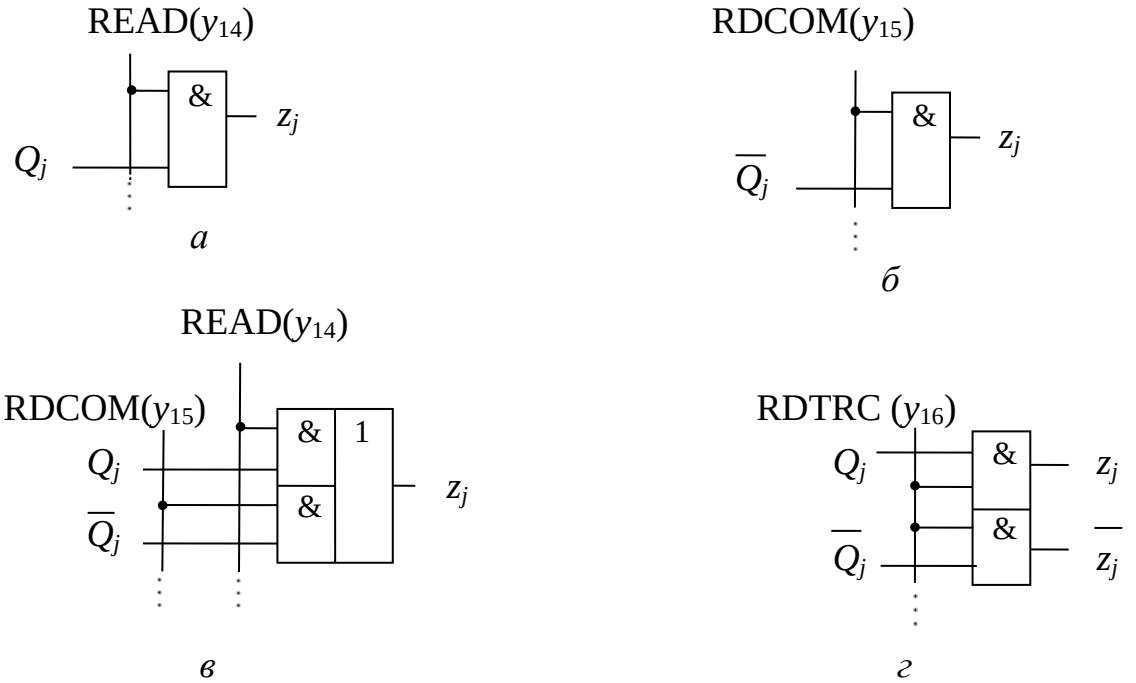
$$z_j = y_{14} Q_j ,$$

RDCOM (y_{15}) – видача слова в інверсному коді (рис. 3.2, б):

$$z_j = y_{15} \overline{Q}_j ,$$

RDTRC (y_{16}) – видача слова у парафазному коді (рис. 3.2, г):

$$\begin{cases} z_j = y_{16} Q_j \\ \overline{z}_j = y_{16} \overline{Q}_j . \end{cases}$$



а *б* *в* *г*

Рис. 3.2. Видача слова з регістра (показано один розряд регістра): *а* – видача слова у прямому коді; *б* – видача слова в інверсному коді; *в* – видача слова у прямому або інверсному коді; *г* – видача слова у парафазному коді.

Методика проектування регістрів

Проектування регістрів на тригерах заданого типу зводиться до синтезу КС, яка формує функції збудження тригерів під час виконання заданих мікрооперацій, а також забезпечує видачу слова з регістра.

Якщо на регістрі виконується одна з мікрооперацій $y_{14} - y_{16}$, то як такий синтез КС відсутній і відповідні КС виглядають як показано на рис. 3.2.

Функції збудження RS-, T-, JK-, та D- тригера подано в табл. 3.1 – 3.4.

Таблиця 3.1. Таблиця функцій збудження RS-тригера

$Q(t_i)$	$Q(t_{i+1})$	F_R	F_S
0	0	×	0
0	1	0	1
1	0	1	0
1	1	0	×

Таблиця 3.2. Таблиця функції збудження T-тригера

$Q(t_i)$	$Q(t_{i+1})$	F_T
0	0	0
0	1	1
1	0	1
1	1	0

Таблиця 3.3. Таблиця функцій збудження JK-тригера

$Q(t_i)$	$Q(t_{i+1})$	F_J	F_K
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

Таблиця 3.4. Таблиця функції збудження D-тригера

$Q(t_i)$	$Q(t_{i+1})$	F_D
0	0	0
0	1	1
1	0	0
1	1	1

Розглянемо чотири випадки синтезу регістрів.

1. Синтез регістра на асинхронних тригерах, на якому виконується одна мікрооперація

Синтез регістра на асинхронних тригерах (асинхронний тригер не має тактового входу) виконується в такій послідовності: беремо таблицю функцій збудження заданого типу тригера; складаємо повну таблицю переходів *i*-го розряду регістра при виконанні заданої мікрооперації, у кожному рядку одержаної таблиці записуємо значення функцій збудження тригера *i*-го розряду, мінімізуємо функції збудження та подаємо їх у відпо-

відній операторній формі (відповідно до заданого елементного базису); будуємо функціональну схему регістра.

Задача 1. Синтезувати 4-розрядний регістр на асинхронних RS-тригерах для виконання на ньому мікрооперації AND(y_4) – кон'юнкція двох слів.

Розв'язування. 1. Беремо таблицю функцій збудження RS-тригера (табл. 3.1):

$Q(t_i)$	$Q(t_{i+1})$	F_R	F_S
0	0	×	0
0	1	0	1
1	0	1	0
1	1	0	×

2. Зазначена мікрооперація задається рівнянням $Q_j(t_{i+1}) = Q_j(t_i)\ \&\ x_j(t_i)$.

3. Складаємо повну таблицю переходів j-го розряду регістра для виконання мікрооперації y_4 (табл. 3.5). При $y_4 = 0$ мікрооперація не виконується, тому у стовпці $Q_j(t_{i+1})$ для кожного рядка повторюємо значення $Q_j(t_i)$. При $y_4 = 1$ стовпець $Q_j(t_{i+1})$ заповнюємо у відповідності з виразом $Q_j(t_{i+1}) = Q_j(t_i)\ \&\ x_j(t_i)$.

Таблиця 3.5. Повна таблиця переходів j-го розряду асинхронного регістра при виконанні мікрооперації AND(y_4)

y_4	$x_j(t_i)$	$Q_j(t_i)$	$Q_j(t_{i+1})$	F_{R_j}	F_{S_j}
0	0	0	0	×	0
0	0	1	1	0	×
0	1	0	0	×	0
0	1	1	1	0	×
1	0	0	0	×	0
1	0	1	0	1	0
1	1	0	0	×	0
1	1	1	1	0	×

4. Керуючись таблицею функцій збудження RS-тригера, заповнюємо стовпці F_{R_j}, F_{S_j} табл. 3.5, наприклад:

фрагмент таблиці функцій збудження RS-тригера

$Q_j(t_i)$	$Q_j(t_{i+1})$	F_R	F_S
0	1	0	1

$\Rightarrow$ . . .

фрагмент таблиці переходів j-го розряду регістра

$Q_j(t_i)$	$Q_j(t_{i+1})$	F_{R_j}	F_{S_j}
0	1	0	1

5. Виконуємо мінімізацію функцій F_{R_j} та F_{S_j}:

y_4 \ $x_j(t_i)$				
	×	0	1	×
	×	0	0	×

$\overline{Q_j(t_i)}$ ← F_{R_j}

$F_{R_j} = y_4 \overline{x}_j$,

y_4 \ $x_j(t_i)$				
	0	×	0	0
	0	×	×	0

$\overline{Q_j(t_i)}$ ← F_{S_j}

$F_{S_j} = 0$.

6. Будуємо схему *j*-го розряду регістра (рис. 3.3).

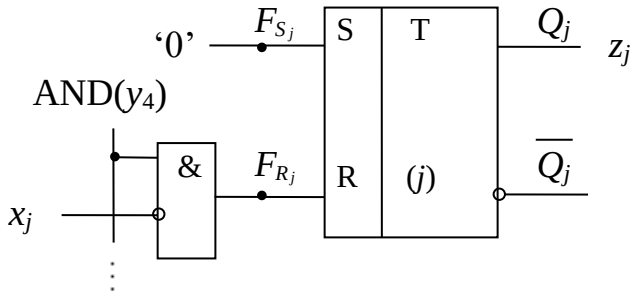


Рис. 3.3. Функціональна схема *j*-го розряду асинхронного регістра, на якому виконується мікрооперація AND(y_4)

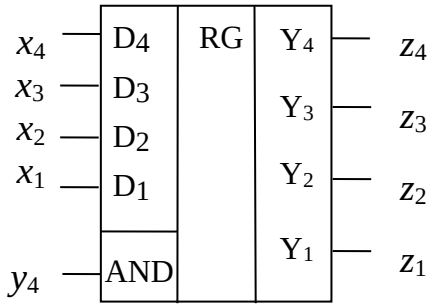


Рис. 3.4. Умовне графічне позначення 4-розрядного асинхронного регістра, на якому виконується мікрооперація AND(y_4)

7. Будуємо функціональну схему регістра (рис. 3.5) (всі розряди регістра ідентичні) та подаємо його умовне графічне позначення (рис. 3.4).

Якщо аргументом функцій збудження тригера *j*-го розряду є $Q_j(t_i)$, $\overline{Q}_j(t_i)$, то тригери мають бути з внутрішньою затримкою (керовані перепадом (фронтом) синхросигналу).

2. Синтез регістра на асинхронних тригерах, на якому виконуються декілька мікрооперацій

У випадку побудови регістра на асинхронних тригерах, на якому виконуються декілька мікрооперацій, узагальнені функції збудження тригера є диз'юнкцією однойменних функцій збудження, що відповідають окремим мікрооперація.

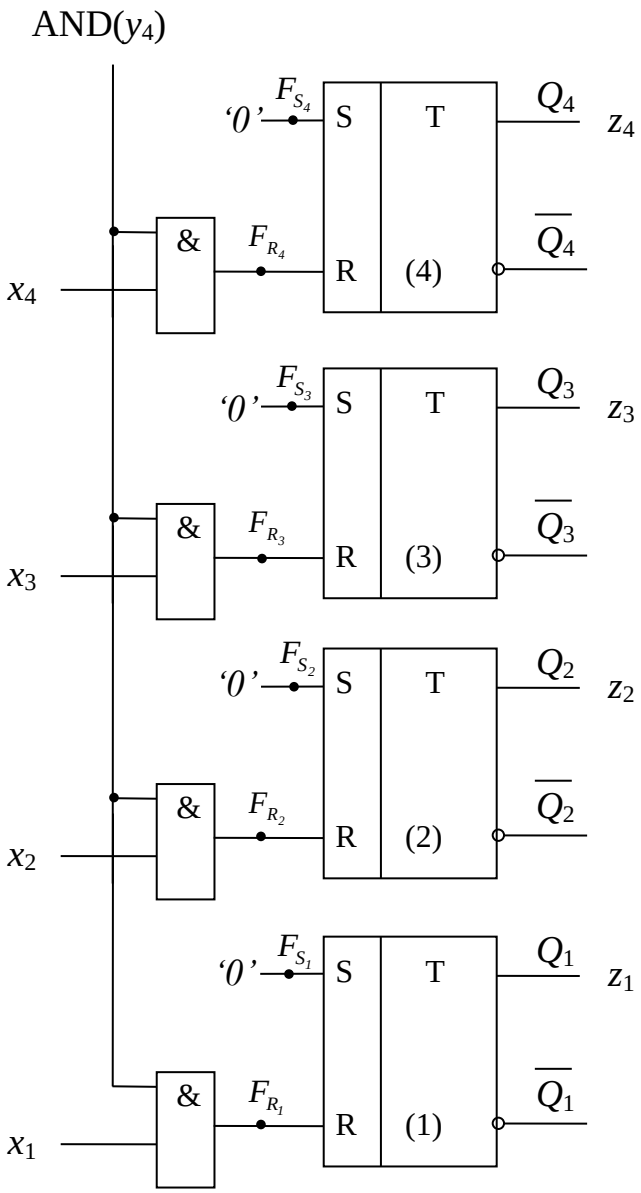


Рис. 3.5. Функціональна схема 4-розрядного асинхронного регістра на RS-тригерах, на якому виконується мікрооперація AND(y_4)

Задача 2. На асинхронних Т-тригерах синтезувати 4-розрядний регістр для виконання на ньому трьох мікрооперацій: WRITE(y_3) – запис слова в регістр, OR(y_5) – порозрядна диз'юнкція двох слів, READ(y_{14}) – видача слова у прямому коді.

Розв'язування. 1. Беремо таблицю функцій збудження Т-тригера (табл. 3.2):

$Q(t_i)$	$Q(t_{i+1})$	F_T
0	0	0
0	1	1
1	0	1
1	1	0

2. Для мікрооперації WRITE(y_3) $Q_j(t_{i+1}) = x_j(t_i)$.

3. Користуючись таблицею функції збудження Т-тригера складаємо повну таблицю переходів j-го розряду регістра для виконання мікрооперації WRITE(y_3) (табл. 3.6).

Таблиця 3.6. Повна таблиця переходів j-го розряду асинхронного регістра при виконанні мікрооперації WRITE(y_3)

y_3	$x_j(t_i)$	$Q_j(t_i)$	$Q_j(t_{i+1})$	$F_{T_j}(y_3)$
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	1	0
1	0	0	0	0
1	0	1	0	1
1	1	0	1	1
1	1	1	1	0

Мінімізуємо функцію збудження $F_{T_j}(y_3)$:

$F_{T_j}(y_3) = y_3\overline{x}_j Q_j \vee y_3 x_j \overline{Q}_j$.

4. Складаємо повну таблицю переходів j-го розряду регістра для виконання мікрооперації OR(y_5) (табл. 3.7), яка описується рівнянням

$$Q_j(t_{i+1}) = Q_j(t_i) \vee x_j(t_i).$$

Таблиця 3.7. Повна таблиця переходів j-го розряду асинхронного регістра при виконанні мікрооперації OR(y_5)

y_5	$x_j(t_i)$	$Q_j(t_i)$	$Q_j(t_{i+1})$	$F_{T_j}(y_5)$
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	0
1	1	0	1	1
1	1	1	1	0

У цьому випадку функція збудження j-го розряду регістра має вигляд $F_{T_j}(y_5) = y_5 x_j \overline{Q}_j$.

5. Знаходимо узагальнену функцію збудження j-го розряду регістра $F_{T_j} = F_{T_j}(y_3) \vee F_{T_j}(y_5) = y_3 \bar{x}_j Q_j \vee y_3 x_j \bar{Q}_j \vee y_5 x_j \bar{Q}_j$.

6. Для мікрооперації READ(y_{14}) $z_j = y_{14} Q_j$.

7. Будуємо схему j-го розряду регістра (рис. 3.6).

8. Будуємо функціональну схему регістра (рис. 3.8) та подаємо його умовне графічне позначення (рис. 3.7).

У конкретний момент часу t_i на регістрі може виконуватись лише одна мікрооперація, в інший момент часу – інша мікрооперація. Виконання двох або більше мікрооперацій в один і той самий момент часу неможливе.

3. Синтез регістра на синхронних тригерах, на якому виконується одна мікрооперація

Якщо на регістрі виконується лише одна мікрооперація, то у повну таблицю переходів j-го розряду регістра не включають значення сигналу мікрооперації y_i. Цей сигнал подають безпосередньо на тактові входи всіх тригерів, він виконує функцію синхросигналу.

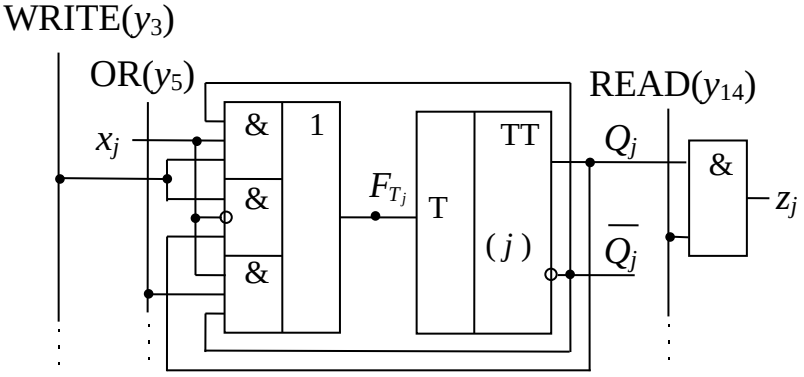


Рис. 3.6. Функціональна схема j-го розряду асинхронного регістра, на якому виконуються мікрооперації WRITE(y_3), OR(y_5) та READ(y_{14})

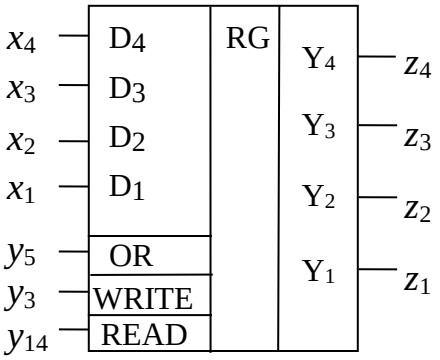


Рис. 3.7. Умовне графічне позначення 4-розрядного асинхронного регістра, та якому виконуються мікрооперації WRITE(y_3), OR(y_5) та READ(y_{14})

Задача 3. Побудувати 4-розрядний регістр на синхронних D-тригерах для реалізації на ньому мікрооперації XOR(y_6), використовуючи для синтезу КС елементи 2I-НЕ .

Розв'язування. 1. Беремо таблицю функцій збудження D-тригера (табл. 3.4):

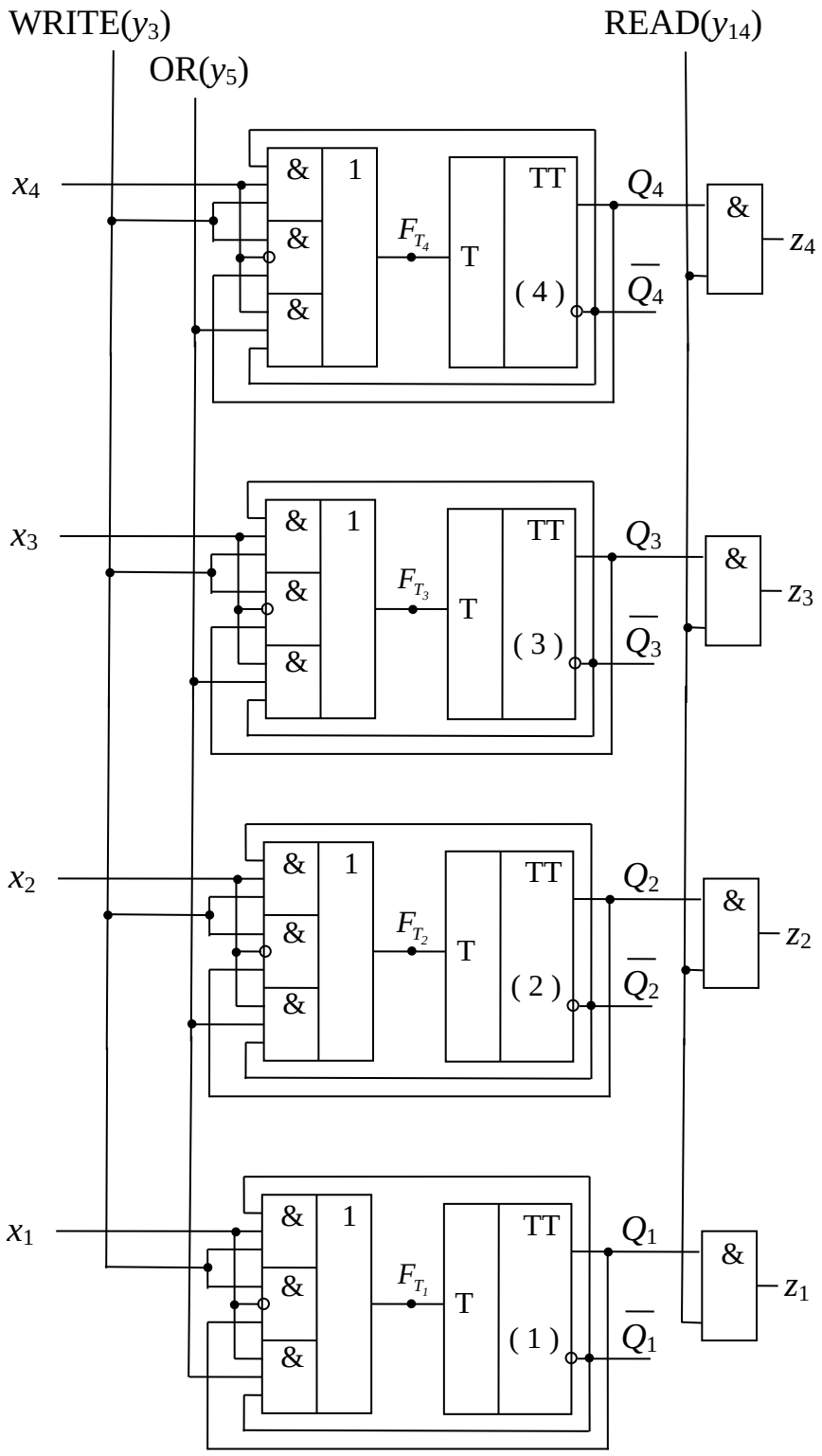


Рис. 3.8. Функціональна схема 4-розрядного асинхронного регістра, на якому виконуються мікрооперації WRITE(y_3), OR(y_5) та READ(y_{14})

$Q_j(t_i)$	$Q_j(t_{i+1})$	F_D
0	0	0
0	1	1
1	0	0
1	1	1

2. Мікрооперація XOR(Y_6) описується рівнянням

$$Q_j(t_{i+1}) = Q_j(t_i) \oplus x_j(t_i).$$

3. Складаємо повну таблицю переходів j-го розряду регістра (табл. 3.8).

Таблиця 3.8. Повна таблиця переходів j-го розряду синхронного регістра при виконанні мікрооперації XOR(y_6)

$x_j(t_i)$	$Q_j(t_i)$	$Q_j(t_{i+1})$	F_{Di}
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

4. Керуючись таблицею функцій збудження D-тригера заповнюємо стовпець F_{D_j} табл. 3.8.

5. Мінімізуємо функцію збудження F_{D_j}:

$x_j(t_i)$ \ $Q_j(t_i)$	0	1
1	1	0

$$F_{D_j} = x_j\overline{Q}_j \vee \overline{x}_j Q_j = \overline{\overline{x_j\overline{Q}_j \vee \overline{x}_j Q_j}} = \overline{\overline{x_j\overline{Q}_j} \cdot \overline{\overline{x}_j Q_j}}\,.$$

6. Будуємо схему j-го розряду регістра (рис. 3.9).

Далі слід побудувати функціональну схему регістра на 4 розряди (всі розряди ідентичні) (самостійно) та навести умовне графічне позначення регістра (рис. 3.10).

4. Синтез регістра на синхронних тригерах, на якому виконуються декілька мікрооперацій

Якщо на регістрі, побудованому на основі синхронних тригерів, виконуються декілька мікрооперацій, то синтез КС залишається таким самим, як і у випадку асинхронних тригерів (для кожної мікрооперації функції збудження тригерів знаходять окремо, а потім об'єднують їх знаком диз'юнкції).

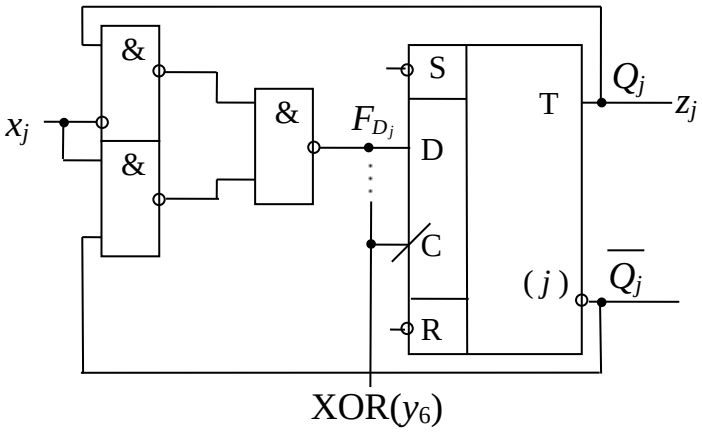


Рис. 3.9. Функціональна схема *j*-го розряду синхронного регістра, на якому виконується мікрооперація XOR (y_6)

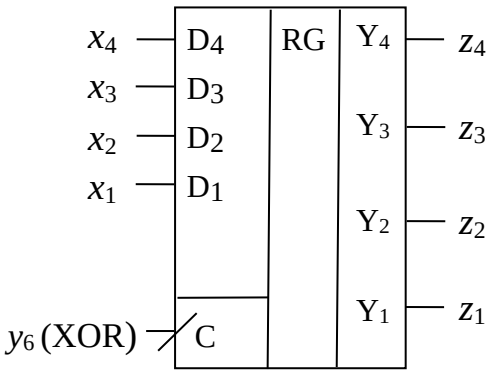


Рис. 3.10. Умовне графічне позначення 4-розрядного синхронного регістра, на якому виконується мікрооперація XOR (y_6)

Задача 4. Побудувати 8-розрядний регістр на синхронних RS-тригерах, на якому виконуються 4 мікрооперації : RESET(y_1) – встановлення в ‘0...0’ регістра, SLL(y_8) – логічний зсув вліво на 1 розряд, SRL(y_9) – логічний зсув вправо на 1 розряд, а також RDTRC(y_{16}) – видача слова у парафазному коді.

Розв’язування. 1. Беремо таблицю функцій збудження RS-тригера (табл. 3.1):

$Q_j(t_i)$	$Q_j(t_{i+1})$	F_R	F_S
0	0	×	0
0	1	0	1
1	0	1	0
1	1	0	×

2. Для реалізації мікрооперації RESET(y_1) на асинхронні входи $\overline{R}$ всіх тригерів регістра слід подавати логічний нуль.

3. Для реалізації мікрооперації RDTRC(y_{16}) КС не синтезуємо, у цьому випадку лише визначимо функції виходів регістра

$$z_j = y_{16}\, Q_j, \quad \overline{z}_j = y_{16}\, \overline{Q}_j$$

та скористаємось схемою на рис. 3.2, г.

4. Складаємо повну таблицю переходів тригера j-го розряду регістра при виконанні мікрооперації SLL(y_8) (табл. 3.9), беручи до уваги, що мікрооперація y_8 задається системою:

$$\begin{cases} Q_j(t_{i+1}) = Q_{j-1}(t_i),\ j \neq 1 \\ Q_1(t_{i+1}) = DR, \end{cases}$$

де DR – вхід для заповнення молодшого розряду регістра під час зсуву.

Таблиця 3.9. Повна таблиця переходів j-го розряду синхронного регістра при виконанні мікрооперації SLL(y_8)

y_8	$Q_{j-1}(t_i)$	$Q_j(t_i)$	$Q_j(t_{i+1})$	$F_{R_j}(y_8)$	$F_{S_j}(y_8)$
0	0	0	0	×	0
0	0	1	1	0	×
0	1	0	0	×	0
0	1	1	1	0	×
1	0	0	0	×	0
1	0	1	0	1	0
1	1	0	1	0	1
1	1	1	1	0	×

При $y_8 = 0$ $Q_j(t_{i+1}) = Q_j(t_i)$, оскільки мікрооперація не виконується.
При $y_8 = 1$ $Q_j(t_{i+1}) = Q_{j-1}(t_i)$.

5. Мінімізуємо функції $F_{R_j}(y_8)$, $F_{S_j}(y_8)$.

y_8 \ $Q_{j-1}(t_i)$	0	0	1	×
	×	0	0	×

$F_{R_j}(y_8)$; $\overline{Q_j(t_i)}$

$$F_{R_j}(y_8) = y_8\bar{Q}_{j-1},$$

y_8 \ $Q_{j-1}(t_i)$	1	×	0	0
	0	×	×	0

$F_{S_j}(y_8)$; $\overline{Q_j(t_i)}$

$$F_{S_j}(y_8) = y_8Q_{j-1}.$$

6. За аналогією легко записати вирази для функцій збудження тригера j-го розряду регістра при виконанні мікрооперації SRL(y_9):

$$F_{R_j}(y_9) = y_9\bar{Q}_{j+1}, \quad F_{S_j}(y_9) = y_9Q_{j+1}.$$

7. Знаходимо узагальнені функції збудження j-го розряду регістра:

$$F_{R_j} = F_{R_j}(y_8) \vee F_{R_j}(y_9) = y_8\bar{Q}_{j-1} \vee y_9\bar{Q}_{j+1},$$
$$F_{S_j} = F_{S_j}(y_8) \vee F_{S_j}(y_9) = y_8Q_{j-1} \vee y_9Q_{j+1}.$$

8. Будуємо функціональну схему регістра.

Три розряди регістра показано на рис. 3.11, а умовне графічне позначення – на рис. 3.12.

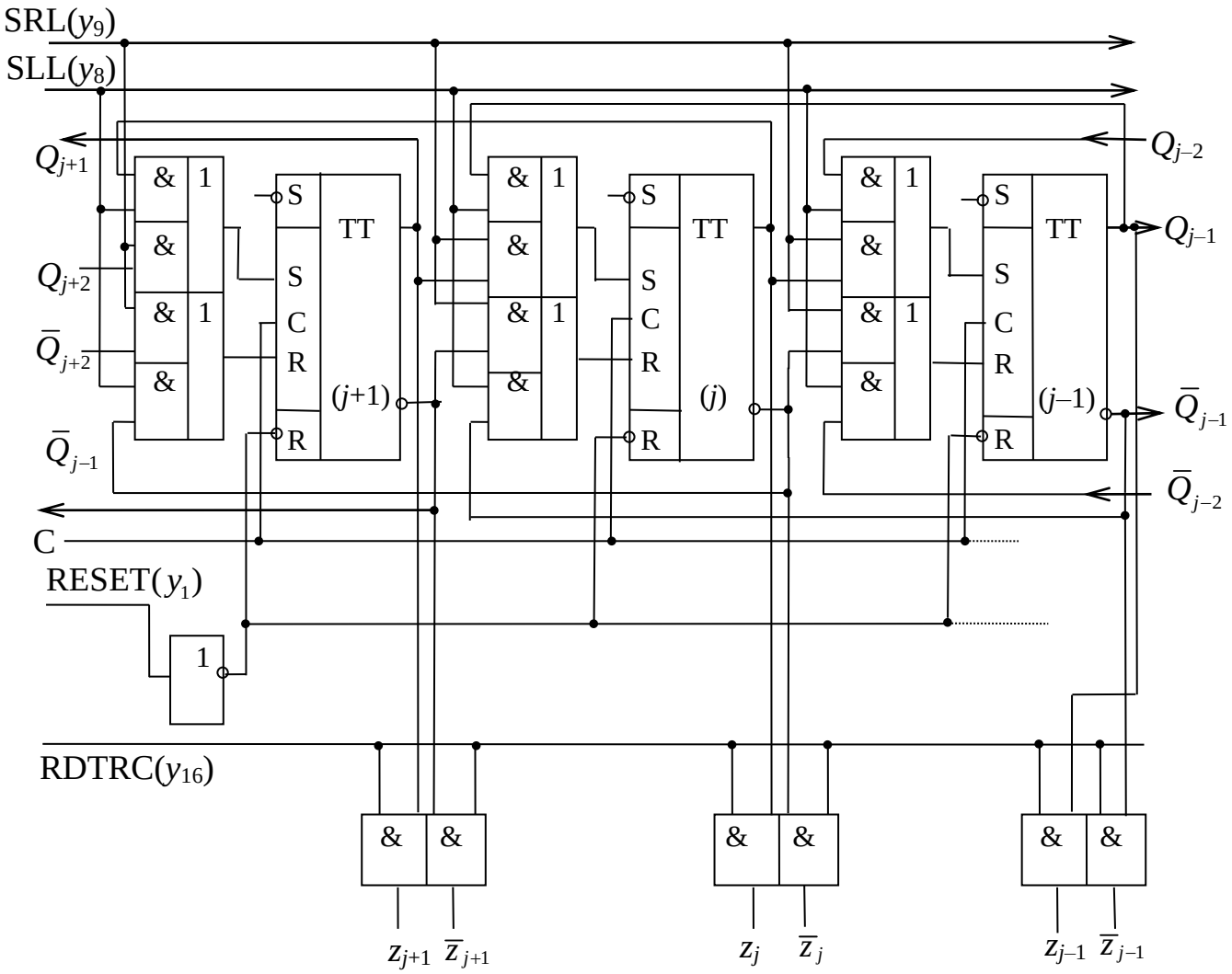


Рис. 3.11. Функціональна схема трьох розрядів синхронного регістра, на якому виконуються мікрооперації RESET, SLL, SRL, RDTRC

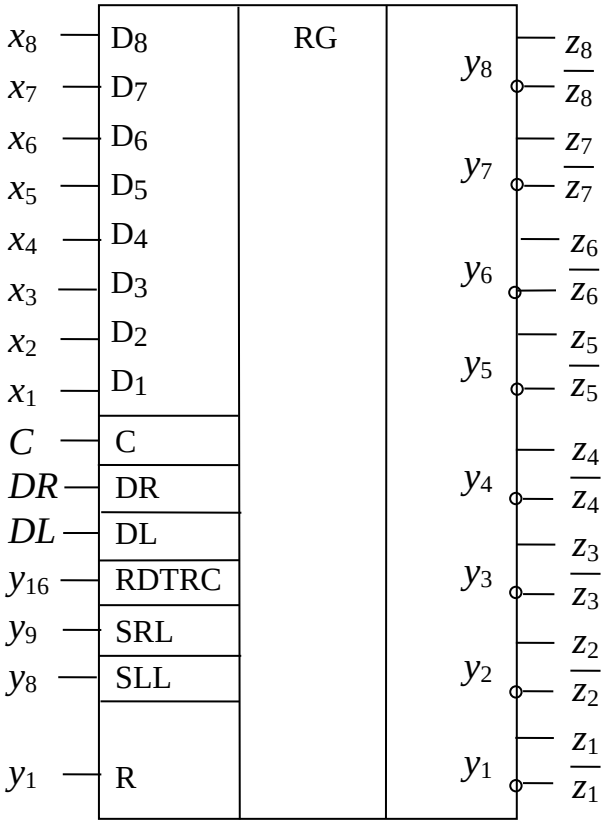
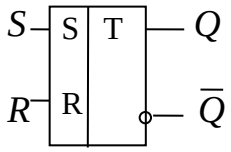
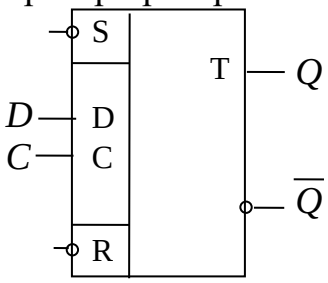
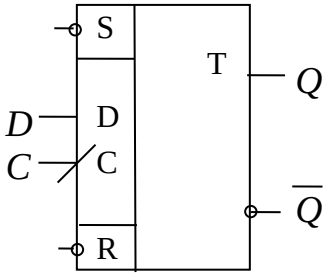
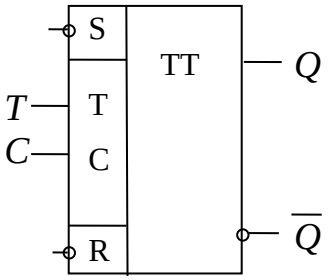
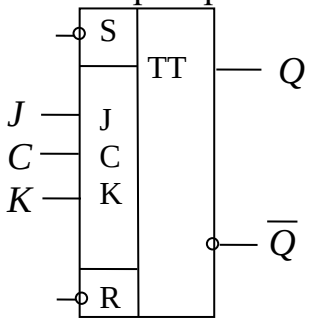


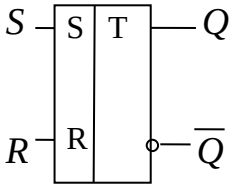
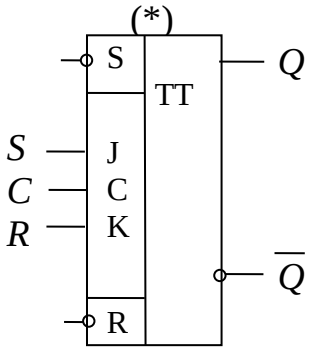
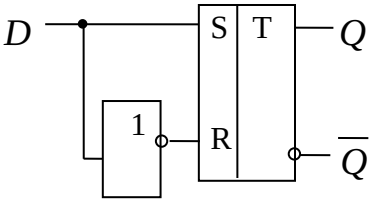
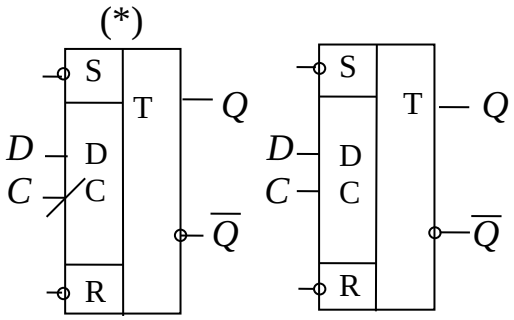
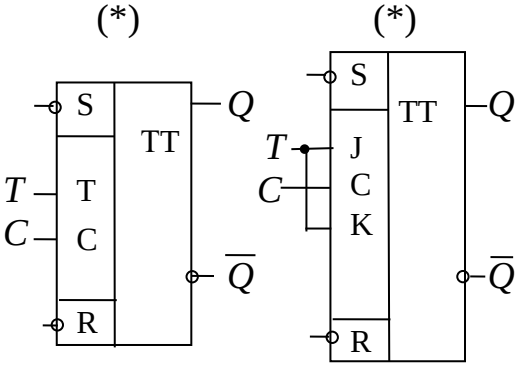
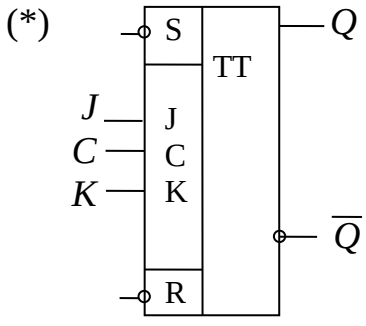
Рис. 3.12. Умовне графічне позначення синхронного 8-розрядного регістра, на якому виконуються мікрооперації RESET, SLL, SRL, RDTRC

Для виконання лабораторної роботи необхідно використовувати тригери з бібліотеки елементів програми ПРОГМОЛС 2.0 (табл. 3.10). З їх допомогою можна одержувати різноманітні типи асинхронних та синхронних тригерів, деякі з яких наведені в табл. 3.11.

Таблиця 3.10. Тригери бібліотеки елементів програми ПРОГМОЛС 2.0

Тип тригера	Позначення
RS-тригер	RS-тригер прозорий  Асинхронний RS-тригер (бістабільна схема на елементах АБО-НЕ)
D-тригер	D-тригер прозорий  Синхронний D- тригер, керований рівнем тактового сигналу D-тригер непрозорий  Синхронний D- тригер, виконаний на основі трьох бістабільних схем (тригер з динамічним записом)
T-тригер	T-тригер  T-тригер, виконаний за MS-схемою
JK-тригер	JK-тригер  JK-тригер, виконаний за MS-схемою

Таблиця 3.11. Асинхронні та синхронні тригери на основі бібліотеки елементів програми ПРОГМОЛС 2.0

Тип тригера	Асинхронний тригер	Синхронний тригер
RS- тригер		
D- тригер		
T- тригер	————	
JK- тригер	————	

Примітка. На тригерах, позначених (*), можна виконувати мікрооперації, аргументами функції збудження яких є Q, $\bar{Q}$. Це тригери з внутрішньою затримкою (керовані перепадом (фронтом) синхросигналу).

Завдання на лабораторну роботу

1. Варіант (дві молодші цифри номера залікової книжки) подати у двійковій системі числення у вигляді слова $a_5a_4a_3a_2a_1$. Побудувати два 4-розрядні регістри на заданих асинхронних тригерах, кожний з яких призначений для виконання однієї мікрооперації

Тип асинхронного тригера	мікрооперація	КС на елементах
RS	OR	2І-НЕ
RS	AND	2АБО-НЕ

2. На асинхронних RS-тригерах побудувати 4-розрядний регістр, призначений для виконання на ньому трьох мікрооперацій: WRITE(y_3), AND(y_4), OR(y_5).

3. Побудувати 4-розрядний регістр на синхронних тригерах, призначений для виконання однієї мікрооперації за варіантом (табл. 3.12).

Таблиця 3.12. Варіанти для виконання п. 3, п. 4. завдання для лабораторної роботи

α_3 α_2 α_1	Пункт 3		Пункт 4	
	тип тригера	мікро-операція	тип тригера	мікро-операції
0 0 0	T-	XOR(y_6)	JK	SET(y_2), AND(y_4), SLL(y_8), READ(y_{14}), WRITE(y_3)
0 0 1	RS-	OR(y_5)	RS	RESET(y_1), AND(y_4), SRL(y_9), RDCOM(y_{15}), WRITE(y_3)
0 1 0	D-	RR(y_{13})	T	SET(y_2), XOR(y_6), SRL(y_9), READ(y_{14}), WRITE(y_3)
0 1 1	T-	SLL(y_8)	RS	RESET(y_1), COM(y_7), RL(y_{12}), RDCOM(y_{15}), WRITE(y_3)
1 0 0	RS-	SRL(y_9)	JK	SET(y_2), COM(y_7), RR(y_{13}), READ(y_{14}), WRITE(y_3)
1 0 1	D-	SLA(y_{10})	RS	RESET(y_1), AND(y_4), RR(y_{13}), RDCOM(y_{15}), WRITE(y_3)
1 1 0	T-	SRA(y_{11})	JK	SET(y_2), OR(y_5), SLL(y_8), READ(y_{14}), WRITE(y_3)
1 1 1	D-	RL(y_{12})	T	SET(y_2), XOR(y_6), SRL(y_9), RDCOM(y_{15}), WRITE(y_3)

4. Побудувати 4-розрядний регістр на синхронних тригерах, на якому виконуються 5 мікрооперацій (за варіантом, табл. 3.12).

5. Для кожного спроектованого тригера (п.1 – п.4 завдання) навести його умовне графічне позначення.

Порядок виконання роботи

1. За завданням викладача побудувати чотири моделі 3-розрядних регістрів: на асинхронних тригерах для виконання однієї мікрооперації (п. 1 завдання); на асинхронних тригерах для виконання трьох мікрооперацій (п. 2 завдання); на синхронних тригерах для виконання однієї мікрооперації (п. 3 завдання); на синхронних тригерах для виконання п'яти мікрооперацій (п. 4 завдання).
2. Дослідити роботу кожної моделі на кількох наборах змінних.
3. Зарисувати цифрові діаграми роботи чотирьох моделей тригерів.

Вимоги до оформлення звіту

Звіт має включати:

1) синтез двох регістрів на асинхронних тригерах, призначених для виконання однієї мікрооперації (п. 1 завдання); регістра на асинхронних тригерах, призначеного для виконання трьох мікрооперацій (п. 2 завдання); регістра на синхронних тригерах, призначеного для виконання однієї мікрооперації (п. 3 завдання); регістра на синхронних тригерах, призначеного для виконання п'яти мікрооперацій (п. 4 завдання), а також умовні графічні позначення цих регістрів;
2) цифрові діаграми роботи моделей чотирьох регістрів, досліджуваних в лабораторії за завданням викладача (по одній з кожного пункту завдання).

Питання для самоперевірки

1. Дати визначення регістра.
2. У чому полягає відмінність тригерів з внутрішньою затримкою і без неї?
3. Що таке довжина регістра?
4. Нарисувати узагальнену структуру регістра.
5. Які регістри називають реверсивними?
6. Від чого залежить складність комбінаційної схеми регістра?
7. За якою методикою здійснюють синтез регістрів, на яких виконується одна мікрооперація, у випадку асинхронних та синхронних тригерів?
8. Чим відрізняється методика проектування регістрів, на яких виконуються декілька мікрооперацій, від методики, коли на регістрах виконується лише одна мікрооперація?
9. Під час виконання яких мікрооперацій синтез комбінаційної схеми регістра відсутній як такий?
10. Навести таблиці функцій збудження RS-, JK-, T-, D-тригера.

11. У яких випадках для побудови регістра необхідно використовувати тригери з внутрішньою затримкою?
12. Чим відрізняється арифметичний зсув слова від логічного зсуву?

Рекомендована література

1. Савельев А.Я. Прикладная теория цифровых автоматов: Учеб. для вузов по спец. ЭВМ. – М. : Высш. шк., 1987. – 344 с.
2. Савельев А.Я. Арифметические и логические основы цифровых автоматов: Учебник – М. : Высш. шк., 1980. – 255 с.
3. Самофалов К.Г., Романкевич А.М., Валуйский В.Н., Каневский Ю.С., Пиневич М.М. Прикладная теория цифровых автоматов. – К. : Вища школа, 1987. – 375 с.
4. Самофалов К.Г., Корнейчук В.И., Тарасенко В.П. Цифровые ЭВМ. Теория и проектирование. – К. : Вища школа, 1989. – 424 с.

ЛАБОРАТОРНА РОБОТА №4. ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ ЛІЧИЛЬНИКІВ НА ПОТЕНЦІАЛЬНИХ ЕЛЕМЕНТАХ

Мета роботи: вивчити різновиди лічильників на потенціальних елементах, опанувати методику їх проектування та дослідження.

Теоретичні відомості

Лічильником називають операційний вузол, призначений для виконання мікрооперації лічби.

Кількість дозволених станів лічильника називають його періодом або модулем.

Залежно від основи системи числення, у якій виконується мікрооперація лічби, розрізняють двійкові, двійково-десяткові, двійково-п’ятіркові лічильники тощо.

Лічильники бувають асинхронні та синхронні.

В асинхронних лічильниках на інформаційні входи асинхронних тригерів (або на тактові входи синхронних тригерів) надходять сигнали з виходів сусідніх розрядів. Тригери в таких лічильниках спрацьовують не одночасно, оскільки перемикання одних тригерів починається лише після зміни стану інших тригерів.

У синхронних лічильниках усі тригери перемикаються одночасно під діянням спільного синхросигналу, що надходить на тактові входи всіх тригерів.

Для побудови лічильників на потенціальних елементах переважно застосовують синхронні тригери з внутрішньою затримкою.

В узагальненій структурі синхронного лічильника на JK-тригерах (рис. 4.1) використовуються такі позначення: КС – комбінаційна схема, CI – лічильний сигнал, $X = (x_n, \ldots, x_1)$ – початковий стан лічильника, $Y = (y_m, \ldots, y_1)$ – сигнали мікрооперацій, $z_n, \ldots, z_1$ – значення розрядів (стан) лічильника, $\Phi_n, \ldots, \Phi_1$ – сигнали функцій міжрозрядних переносів лічильника.

Для встановлення лічильника у стан (0, …, 0); (1, …1) перед виконанням мікрооперації використовують асинхронні входи $\overline{R}$ та $\overline{S}$ тригерів відповідно.

Комбінаційна схема формує функції збудження Φ_j ($j = 1, \ldots, n$) тригерів, що відіграють роль сигналів переносів (Φ_j – перенос в j-й розряд лічильника).

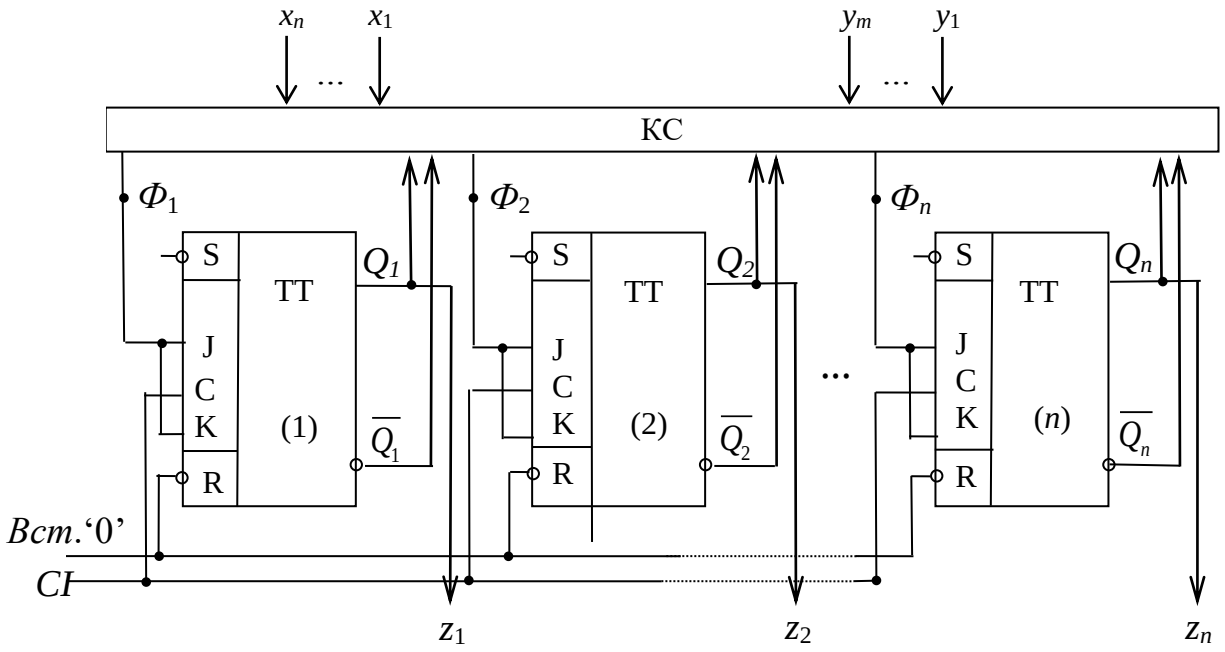


Рис. 4.1. Узагальнена структура синхронного лічильника

Лічильники з природним порядком лічби

За характером мікрооперації лічби лічильники з природним порядком лічби поділяють на інкрементні, декрементні, реверсивні.

У моменти надходження чергового лічильного сигналу стан інкрементного лічильника змінюється на +1, а декрементного на –1. Реверсивний лічильник може виконувати мікрооперацію інкременту або декременту залежно від значення сигналу на вході управління.

Перемикання тригера молодшого розряду лічильника здійснюється з надходженням кожного лічильного сигналу *CI*, а решти тригерів – лише у тому випадку, коли всі тригери молодших розрядів встановлені в 1 (інкрементний лічильник) або в 0 (декрементний лічильник).

За способом організації переносу (позички) між розрядами лічильники з природнім порядком лічби поділяють на такі типи:

- з послідовним переносом,
- з наскрізним переносом,
- з паралельним переносом,
- з груповим переносом.

У лічильниках з ***послідовним переносом*** перенос (позичка) у сусідній старший розряд формується лише після перемикання тригера в попередньому (молодшому) розряді. Такі лічильники є асинхронними. У цьому випадку тригери перемикаються не одночасно (несинхронно).

У випадку ***наскрізного переносу*** кола переносу організують таким чином, щоб значення функції переносу *j*-го (молодшого) розряду лічиль-

ника було аргументом функції переносу j+1 (наступного) розряду. У цьому випадку сигнали переносів для кожного розряду лічильника формуються почергово, починаючи з молодших розрядів.

У лічильниках з ***паралельним (одночасним) переносом*** аргументами функцій збудження для кожного розряду є сигнали на виходах тригерів попередніх (молодших) розрядів. Переноси для всіх розрядів лічильника формуються одночасно.

У лічильниках з ***груповим переносом*** розряди розбивають на групи і в межах однієї групи організують паралельний перенос, а між групами – послідовний або наскрізний.

Тригери n-розрядного інкрементного лічильника з послідовним переносом, побудованого на синхронних Т-тригерах (рис. 4.2), перемикаються за заднім фронтом тактового сигналу (рис. 4.3).

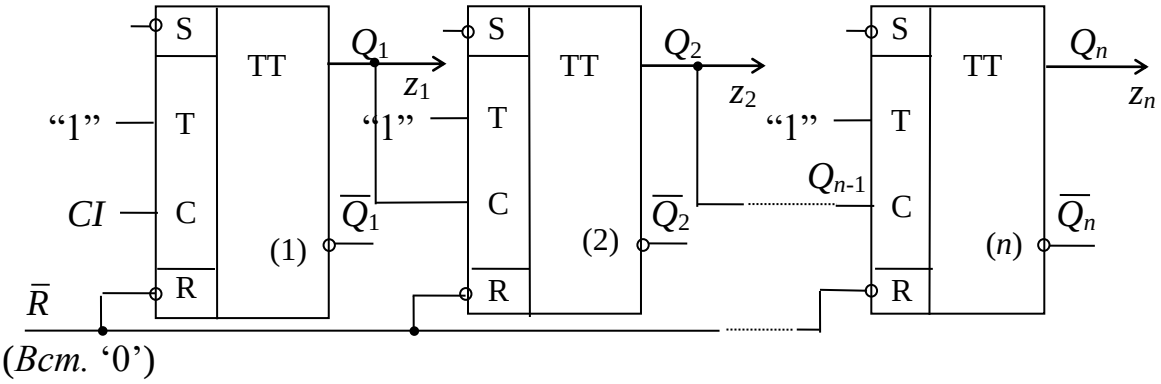


Рис. 4.2. Організація послідовного переносу в лічильниках

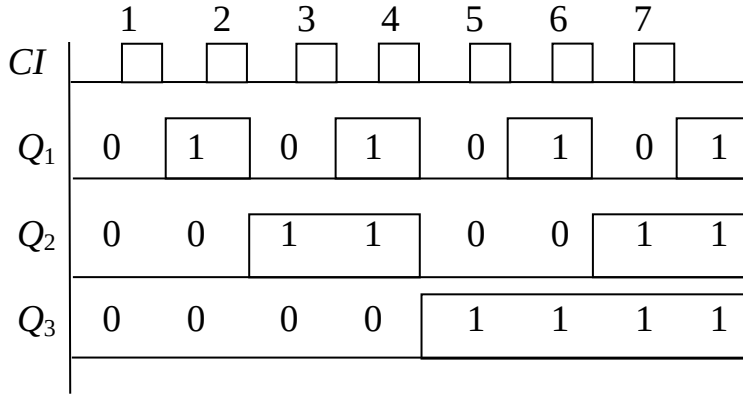


Рис. 4.3. Часова діаграма лічильника з послідовним переносом

Якщо ***схема лічильника*** реалізує ***звичайний (природний) порядок лічби*** з модулем 2^n і ***має кола паралельного переносу***, то функції міжрозрядних переносів Φ_j мають такий вигляд.

Для інкрементного лічильника:

$$\begin{cases} \Phi_j = Q_1 \cdot Q_2 \cdot \ldots \cdot Q_{j-1}, \ j = 2, \ldots, n, \\ \Phi_1 = 1 \end{cases}$$

Перемикання тригера *j*-го розряду здійснюється лише тоді, коли всі тригери молодших розрядів встановлені в ‘1’ (рис. 4.4).

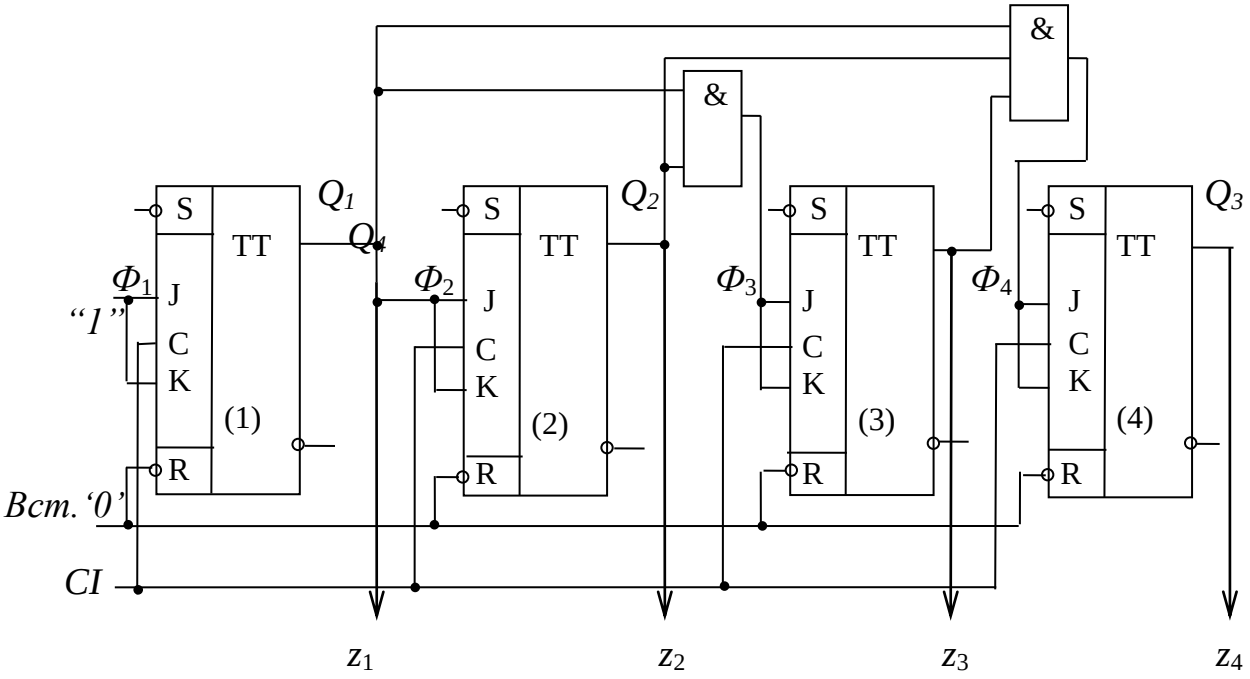


Рис. 4.4. Функціональна схема інкрементного лічильника з паралельним переносом на JK-тригерах

Для декрементного лічильника:

$$\begin{cases} \Phi_j = \bar{Q}_1 \cdot \bar{Q}_2 \cdot \ldots \cdot \bar{Q}_{j-1}, \ \ j = 2, \ldots, n, \\ \Phi_1 = 1 \end{cases}$$

У декрементному лічильнику на входи елементів І (див. рис. 4.4) подаються сигнали з інверсних виходів $\bar{Q}_j$ тригерів . Перемикання тригера *j*-го розряду здійснюється лише тоді, коли всі тригери молодших розрядів встановлені в ‘0’.

Для реверсивного лічильника:

$$\begin{cases} \Phi_j = Q_1 \cdot Q_2 \cdot \ldots \cdot Q_{j-1} y_1 \vee \bar{Q}_1 \cdot \bar{Q}_2 \cdot \ldots \cdot \bar{Q}_{j-1} y_2, \ j = 2, \ldots, n, \\ \Phi_1 = 1 \end{cases}$$

При $y_1 = 1$ виконується мікрооперація інкременту, а при $y_2 = 1$ – декременту.

Для ***лічильників з наскрізним переносом*** функції міжрозрядних переносів Φ_j мають такий вигляд.

Для інкрементного лічильника:

$$\begin{cases} \Phi_j = \Phi_{j-1} \cdot Q_{j-1}, \quad j = 2, ..., n, \\ \Phi_1 = 1 \end{cases}$$

Для декрементного лічильника:

$$\begin{cases} \Phi_j = \Phi_{j-1} \bar{Q}_{j-1}, \quad j = 2, ..., n, \\ \Phi_1 = 1 \end{cases}$$

Для реверсивного лічильника:

$$\begin{cases} \Phi_1 = 1, \\ \Phi_2 = \Phi'_2 \vee \Phi''_2, \text{ де } \Phi'_2 = y_1 \cdot Q_1, \ \Phi''_2 = y_2 \cdot \bar{Q}_1 \\ \Phi_3 = \Phi'_3 \vee \Phi''_3, \text{ де } \Phi'_3 = \Phi'_3 \cdot Q_2, \ \Phi''_3 = \Phi''_2 \cdot \bar{Q}_2, \\ \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\ \Phi_j = \Phi'_j \vee \Phi''_j, \text{ де } \Phi'_j = \Phi'_{j-1} \cdot Q_{j-1}, \ \Phi''_j = \Phi''_{j-1} \cdot \bar{Q}_{j-1}, \\ \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\ \Phi_n = \Phi'_n \vee \Phi''_n, \text{ де } \Phi'_n = \Phi'_{j-1} \cdot Q_{n-1}, \ \Phi''_n = \Phi''_{n-1} \cdot \bar{Q}_{n-1}. \end{cases}$$

Приклад реверсивного лічильника з наскрізним переносом з модулем 16 ($n = 4$) наведено на рис. 4.5.

Лічильники з наскрізним переносом поступаються лічильникам з паралельним переносом швидкодією.

Приклад реалізації групового переносу наведено на рис. 4.6. У межах групи реалізовано паралельний перенос, а між групами – наскрізний (C_4 – перенос з молодшої тетради у старшу, C_8 – вихідний перенос лічильника).

Основними часовими характеристиками лічильників є :

$f_{\text{ліч}}$ – максимальна частота надходження лічильних сигналів;

$t_{\text{п}}$ – час переходу лічильника з одного стану в інший.

Параметр $f_{\text{ліч}}$ характеризує мінімально допустимий період надходження лічильних сигналів, при якому лічильник працює без збоїв.

Час переходу $t_{\text{п}}$ лічильника з одного стану в інший – це інтервал часу між моментами надходження лічильних сигналів, при якому лічильник без збоїв переходить з одного стану в інший.

Параметр $t_{\text{п}}$ залежить від типу тригера та організації переносу між розрядами лічильника.

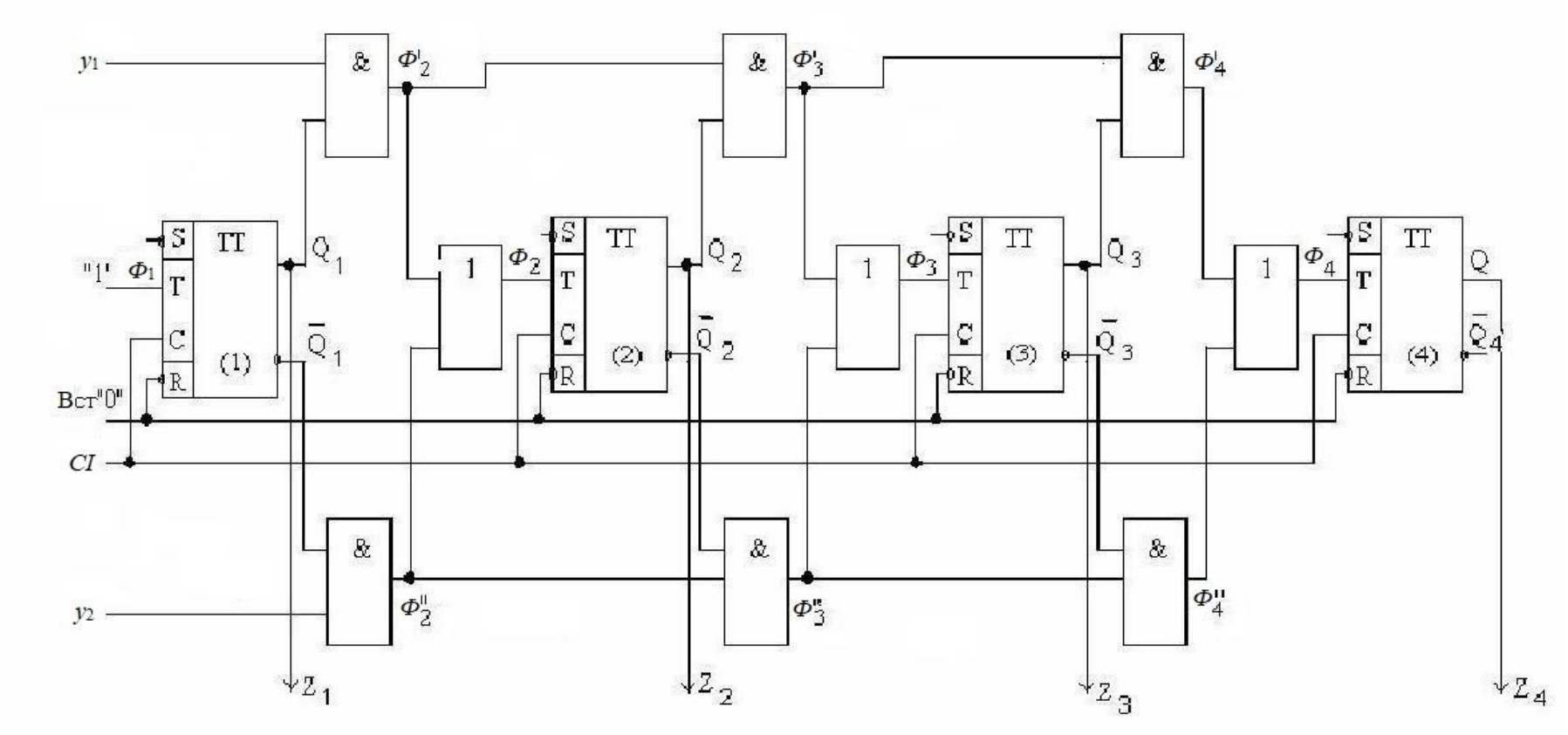


Рис. 4.5. Функціональна схема реверсивного лічильника з наскрізним переносом на Т-тригерах

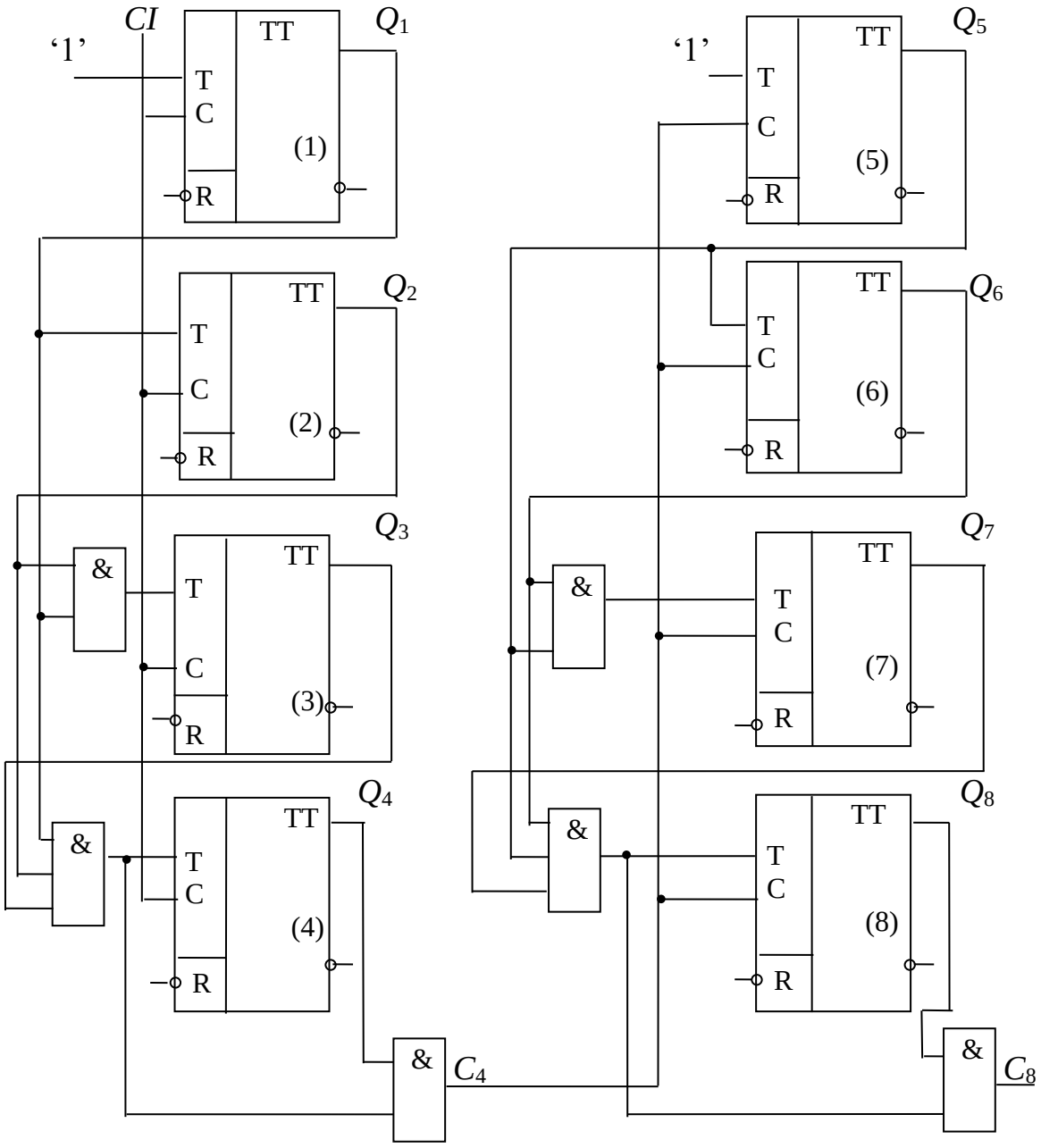


Рис. 4.6. Реалізація групового переносу в лічильнику

Лічильники з довільним (неприродним) порядком лічби

Синхронний лічильник з довільним порядком лічби (періодом) синтезують за такою методикою.

1. Визначити кількість розрядів лічильника

$n = \lceil \log_2 M \rceil$, де M – період (модуль) лічильника.

2. Взяти таблицю переходів та таблицю функцій збудження заданого типу тригера.

3. Скласти таблицю переходів лічильника та знайти функції збудження тригерів для кожного розряду лічильника.

4. Мінімізувати функції збудження тригерів.

5. Побудувати функціональну схему лічильника.

Функції збудження JK-, T- та D-тригерів подані в табл. 4.1.

Таблиця 4.1. Функції збудження тригерів

$Q(t_i)$	$Q(t_{i+1})$	JK-тригер		T-тригер	D-тригер
		F_J	F_K	F_T	F_D
0	0	0	×	0	0
0	1	1	×	1	1
1	0	×	1	1	0
1	1	×	0	0	1

Задача. Синтезувати лічильник на T-тригерах, який може перебувати в таких шести станах: 000, 001, 010, 011, 110, 111.

Розв'язування. 1. Знаходимо розрядність лічильника $n = \lceil \log_2 6 \rceil = 3$.

2. Беремо таблицю переходів T-тригера та таблицю функції збудження T-тригера:

таблиця переходів T-тригера

T	$Q(t_{i+1})$
0	$Q(t_i)$
1	$\overline{Q}(t_i)$

функція збудження T-тригера

$Q(t_i)$	$Q(t_{i+1})$	F_T
0	0	0
0	1	1
1	0	1
1	1	0

3. Складемо таблицю переходів лічильника (табл. 4.2).

Таблиця 4.2. Таблиця переходів лічильника

Стани лічильника						Функції міжрозрядних переносів		
$Q_3(t_i)$	$Q_2(t_i)$	$Q_1(t_i)$	$Q_3(t_{i+1})$	$Q_2(t_{i+1})$	$Q_1(t_{i+1})$	Φ_{T_3}	Φ_{T_2}	Φ_{T_1}
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	1	0	1	0	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

У таблиці переходів лічильника записуємо коди станів лічильника до надходження чергового лічильного сигналу (момент часу t_i) і після його надходження (момент часу t_{i+1}), а потім для кожного рядка таблиці, порівнюючи $Q_j(t_i)$ та $Q_j(t_{i+1})$, записуємо значення Φ_{T_j} (j = 1, 2, 3) міжрозрядних переносів.

4. Виконаємо мінімізацію функцій Φ_{T_j}. З табл. 4.2 визначаємо $\Phi_{T_1} = 1$, а для Φ_{T_2}, Φ_{T_3} побудуємо діаграми Вейча (стани 100 та 101 в лічильнику відсутні, тому на діаграмі позначимо їх ×):

$Q_3(t_i)$ \ $Q_2(t_i)$	0	1	×	×
0	0	0	1	0

$Q_1(t_i)$ Φ_{T_2}

$Q_3(t_i)$ \ $Q_2(t_i)$	0	1	×	×
0	0	1	0	0

$Q_1(t_i)$ Φ_{T_3}

$$\Phi_{T_2} = Q_3Q_1 \vee \bar{Q}_2Q_1, \qquad \Phi_{T_3} = Q_1Q_2.$$

5. Будуємо функціональну схему лічильника (рис. 4.7).

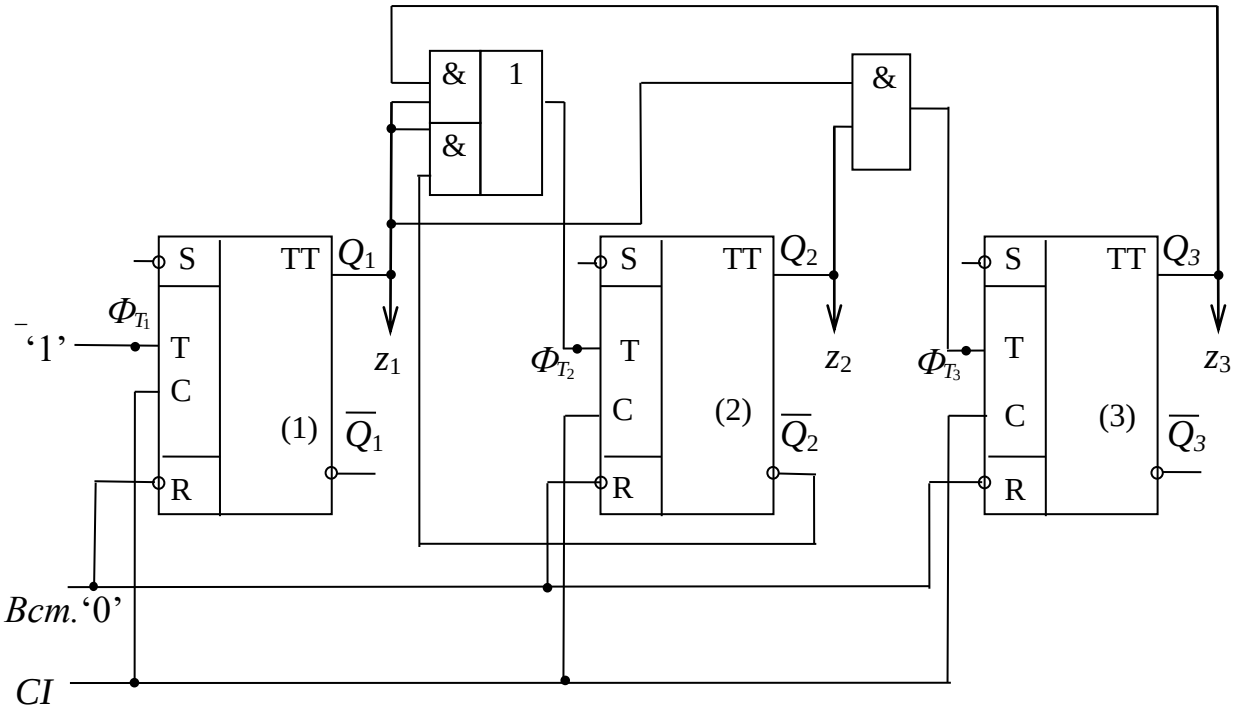


Рис. 4.7. Функціональна схема лічильника з періодом M = 6

Кільцеві лічильники

Кільцеві лічильники будують на регістрах зсуву.

Порядок лічби кільцевого лічильника задають функцією лічби (формулою) Φ.

До складу лічильника входить регістр зсуву та комбінаційна схема (КС) (рис. 4.8). Регістр зсуву має вхід DL – для заповнення розряду (1-го), що звільняється під час зсуву вправо, вхід DR – для заповнення розряду (n-го), що звільняється, під час зсуву вліво.

Виходи Q_j тригерів (розрядів регістра) подають на комбінаційну схему, яка реалізує задану функцію лічби Φ. Вихід КС подають на перший розряд регістра зсуву (при зсуві вправо) або на n-й розряд – при зсуві вліво.

Період (модуль) кільцевого лічильника залежить від початкового стану регістра зсуву.

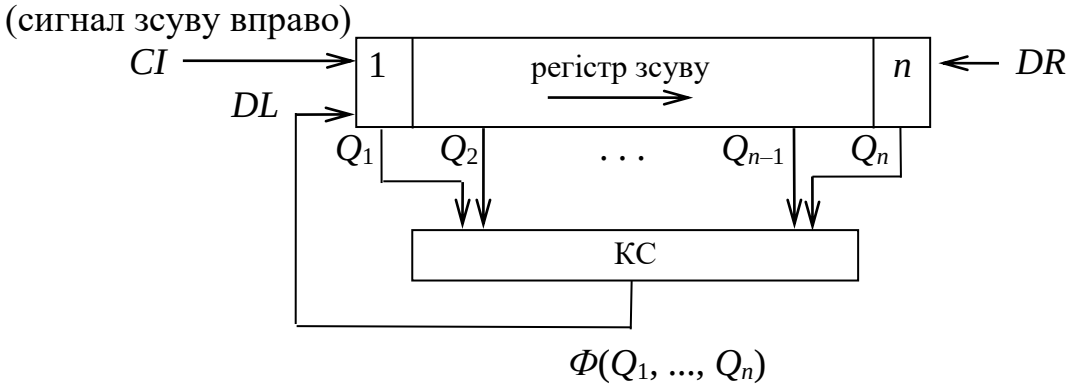


Рис. 4.8. Узагальнена структура кільцевого лічильника (із зсувом вправо)

Якщо функція лічби $\Phi(Q_1, ..., Q_n)$ лінійна, то кількість станів кільцевого лічильника дорівнює 2^n.

Нехай $\Phi = Q_1 \oplus Q_3 \oplus Q_5$. Функціональна схема 5-розрядного кільцевого лічильника, що реалізує цю функцію лічби Φ, зображена на рис. 4.9, а його стани подано в табл. 4.3.

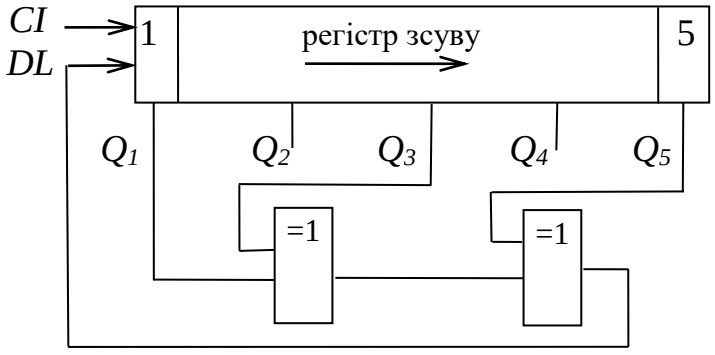


Рис. 4.9. Кільцевий лічильник, що реалізує функцію лічби $\Phi = Q_1 \oplus Q_3 \oplus Q_5$

Якщо $\Phi(Q_1, ..., Q_n)$ – нелінійна функція, то після певної кількості лічильних сигналів (зсувів) лічильник може перейти у такий стан, після якого, незважаючи на надходження нових лічильних сигналів, його стан змінюватися не буде, або перейти до циклічного повторення кількох станів.

Таблиця 4.3. Таблиця станів кільцевого лічильника

Номер періоду	Стани лічильника Q_1 Q_2 Q_3 Q_4 Q_5	Кількість станів у періоді
1	**0 0 0 0 0**	1
2	**0 0 0 0 1**	15
	1 0 0 0 0	
	1 1 0 0 0	
	1 1 1 0 0	
	0 1 1 1 0	
	1 0 1 1 1	
	1 1 0 1 1	
	0 1 1 0 1	
	0 0 1 1 0	
	1 0 0 1 1	
	0 1 0 0 1	
	1 0 1 0 0	
	0 1 0 1 0	
	0 0 1 0 1	
	0 0 0 1 0	
3	**0 0 1 0 0**	15
	1 0 0 1 0	
	1 1 0 0 1	
	0 1 1 0 0	
	1 0 1 1 0	
	0 1 0 1 1	
	1 0 1 0 1	
	1 1 0 1 0	
	1 1 1 0 1	
	1 1 1 1 0	
	0 1 1 1 1	
	0 0 1 1 1	
	0 0 0 1 1	
	1 0 0 0 1	
	0 1 0 0 0	
4	**1 1 1 1 1**	1

Наприклад, якщо КС реалізує функцію лічби

$$\Phi = Q_2 \vee Q_4,$$

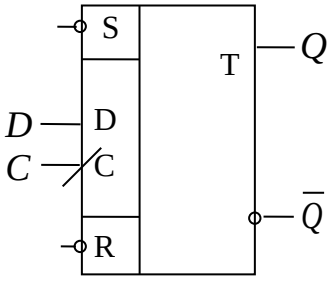
а початковий стан дорівнює 00010, то після надходження п’яти лічильних сигналів лічильник циклічно повторюватиме стани 01010 та 10101 (табл. 4.4).

Таблиця 4.4. Таблиця станів кільцевого лічильника, що реалізує функцію $\Phi = Q_2 \vee Q_4$, з початковим станом 00010

Номер лічильного сигналу	Q_1	Q_2	Q_3	Q_4	Q_5
Поч. стан	**0**	**0**	**0**	**1**	**0**
1	1	0	0	0	1
2	0	1	0	0	0
3	1	0	1	0	0
4	0	1	0	1	0
5	1	0	1	0	1
6	0	1	0	1	0
7	1	0	1	0	1
8	0	1	0	1	0
9	1	0	1	0	1
10	0	1	0	1	0

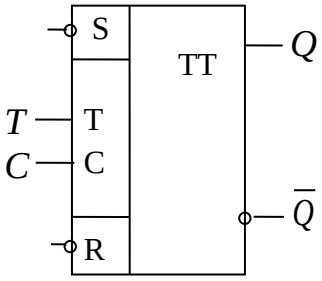
Для виконання лабораторної роботи необхідно використати такі тригери з бібліотеки елементів програми ПРОГМОЛС 2.0 (рис. 4.10):

D-тригер непрозорий



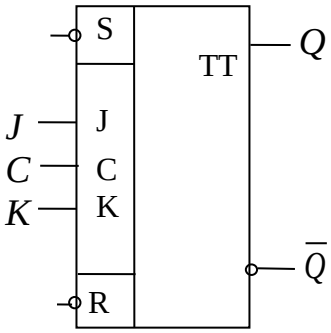
виконаний на основі трьох бістабільних схем

T-тригер



виконаний за MS-схемою

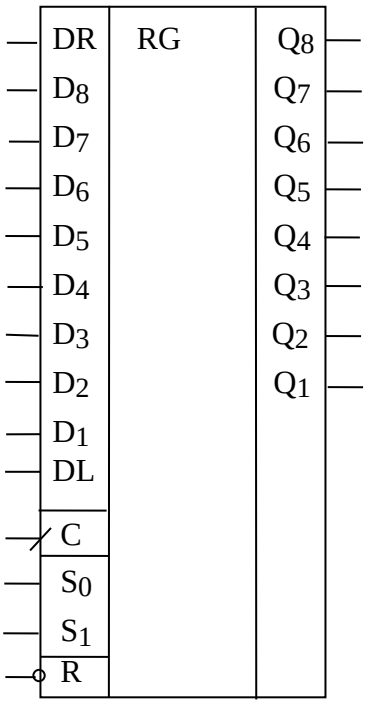
JK-тригер



виконаний за MS-схемою

Рис. 4.10. Тригери бібліотеки програми ПРОГМОЛС 2.0

Кільцевий лічильник необхідно організувати на такому синхронному регістрі зсуву (взяти з бібліотеки) (рис. 4.11):



а

S_0	S_1	Режим
0	0	зберігання
0	1	зсув вправо
1	0	зсув вліво
1	1	запис в регістр

б

Рис. 4.11. Синхронний 8-розрядний регістр зсуву:
а – умовне графічне позначення регістра; *б* – режими роботи регістра

Завдання на лабораторну роботу

1. Варіант (дві молодші цифри номера залікової книжки) подати у двійковій системі числення у вигляді слова $\alpha_5\alpha_4\alpha_3\alpha_2\alpha_1$. Побудувати три лічильники із звичайним порядком лічби з періодом 16:

α_1	тип лічильника
0	- інкрементний лічильник з наскрізним переносом - декрементний лічильник з паралельним переносом - реверсивний лічильник з наскрізним переносом
1	- декрементний лічильник з наскрізним переносом - інкрементний лічильник з паралельним переносом - реверсивний лічильник з паралельним переносом

2. Побудувати лічильник на тригерах заданого типу (за варіантом), який у процесі лічби змінює свої стани за таблицею:

стани лічильника

Q_4	Q_3	Q_2	Q_1
0	0	0	α_1
0	0	1	0
0	0	1	1
0	1	0	α_2
0	1	1	0
1	0	0	α_3
1	0	1	0
1	0	1	1
1	1	0	α_4
1	1	1	0

α_2	α_1	тип тригера
0	0	D-
0	1	T-
1	0	JK-
1	1	RS-

3. Побудувати двійково-десятковий лічильник (2 десяткові цифри (8 двійкових розрядів), скористатися методикою проектування лічильників з неприродним порядком лічби).

4. Побудувати 2 кільцеві лічильники, що реалізують функції лічби:

$$\Phi_1 = \alpha_1 \cdot Q_1 \oplus \alpha_2 \cdot Q_2 \oplus \alpha_3 \cdot Q_3 \oplus \alpha_4 \cdot Q_4 \oplus \alpha_5 \cdot Q_5,$$
$$\Phi_2 = \alpha_1 \cdot Q_1 \vee \alpha_2 \cdot Q_2 \vee \alpha_3 \cdot Q_3 \vee \alpha_4 \cdot Q_4 \vee \alpha_5 \cdot Q_5.$$

Порядок виконання роботи

1. За завданням викладача побудувати модель одного з лічильників із звичайним порядком лічби (п. 1 завдання), перевірити його працездатність та дослідити часову діаграму роботи.

2. Побудувати модель лічильника, який змінює свій стан за таблицею (п. 2 завдання), перевірити правильність його функціонування та дослідити часову діаграму роботи.

3. Побудувати модель двійково-десяткового лічильника та дослідити його функціонування (п. 3 завдання).

4. За завданням викладача побудувати модель одного з кільцевих лічильників (п. 4 завдання), дослідити його роботу при кількох різних початкових станах та зарисувати цифрову діаграму роботи.

Вимоги до оформлення звіту

Звіт має включати:

1) схеми трьох лічильників із звичайним порядком лічби (п. 1 завдання);
2) синтез лічильника, що працює за таблицею (п. 2 завдання), та двійково-десяткового лічильника (п. 3 завдання);
3) часові діаграми роботи двох лічильників – із звичайним порядком лічби, досліджуваного за завданням викладача, та лічильника, що працює за таблицею;
4) схеми двох кільцевих лічильників (п. 4 завдання) та цифрові діаграми їх роботи.

Питання для самоперевірки

1. Нарисувати узагальнену структуру лічильника.
2. Як класифікують лічильники з природним порядком лічби за способом організації переносу між розрядами?
3. Навести приклад лічильника з послідовним переносом.
4. Записати функції збудження *j*-го розряду інкрементного, декрементного та реверсивного лічильників з паралельним переносом.
5. Записати функції збудження *j*-го розряду інкрементного, декрементного та реверсивного лічильників з наскрізним переносом.
6. Порівняти між собою паралельний та наскрізний перенос у лічильниках.
7. Назвати основні часові характеристики лічильників.
8. Сформулювати методику синтезу синхронних лічильників з неприродним порядком лічби.
9. Подати функції збудження JK-,T- та D- тригера у вигляді таблиці.
10. Нарисувати узагальнену структуру кільцевого лічильника.
11. Від чого залежить період кільцевого лічильника?

Рекомендована література

1. Рабинович З.Л., Раманаускас В.А. Типовые операции в вычислительных машинах. – К. : Техніка, 1980. – 264 с.
2. Joseph Cavanagh ComputerArithmetic and Verilog HDL Fundamentals. – Santa Clara University, California, USA: CRC Press, 2010. – 952 p.
3. Кривуля Г.Ф., Рябенький В.М., Буряк В.С. Схемотехніка. – Харків: Компанія СМІТ, 2007. – 250 с.
4. Галчёнков О.Н., Долголенко А.Н. Корнейчук В.И. Компьютерная схемотехника и архитектура компьютеров. Учебное пособие. – К. : Корнійчук, 2013. – 604 с.

ЗМІСТ