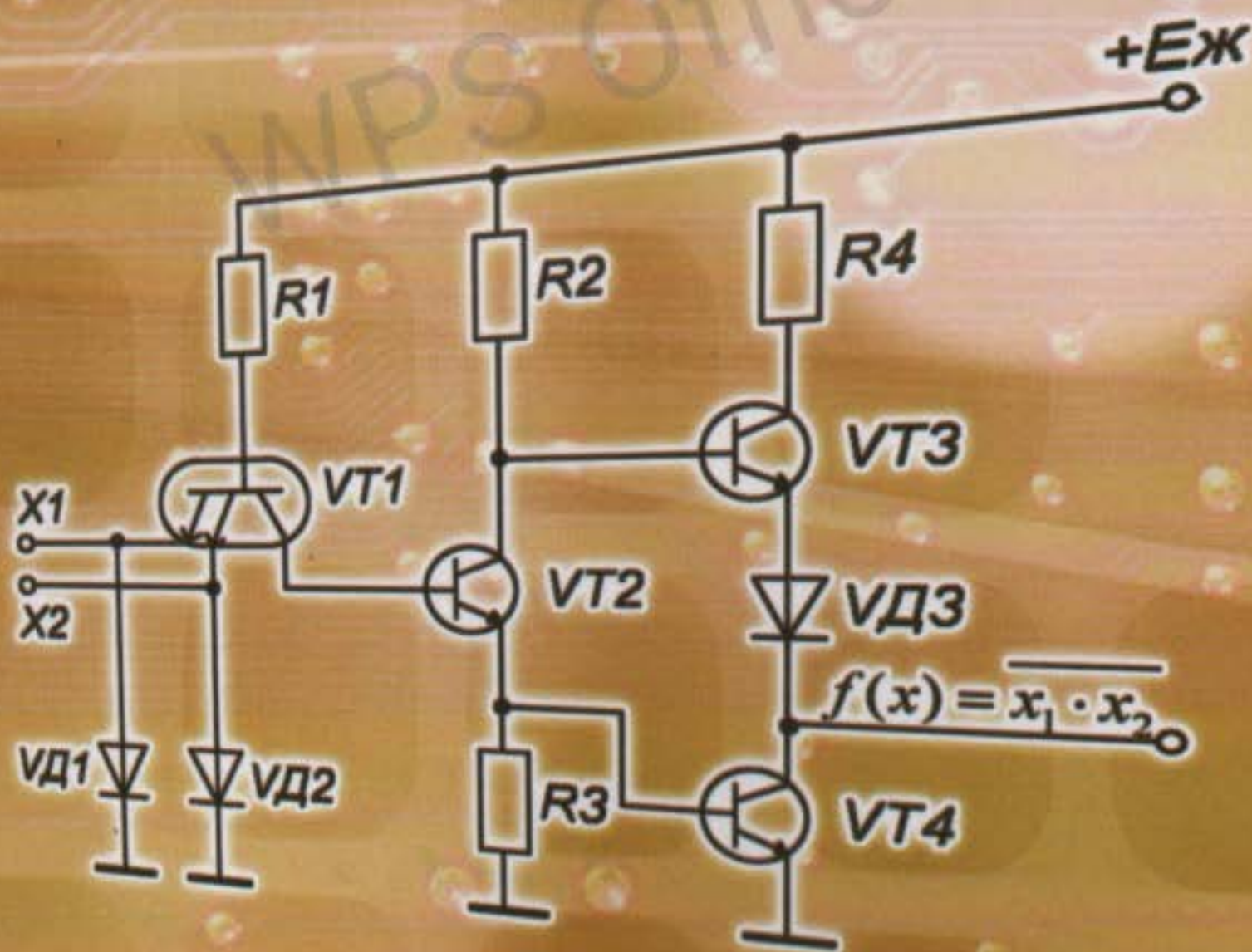


Матвієнко М.П., Розен В.П.

КОМП'ЮТЕРНА СХЕМОТЕХНІКА



МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ

Матвієнко М. П., Розен В. П.

КОМП'ЮТЕРНА СХЕМОТЕХНІКА

*Рекомендовано Міністерством освіти і науки,
молоді та спорту України як навчальний посібник
для студентів вищих навчальних закладів*



Київ-2016

ББК 32.973
УДК 517.1
М 33

Копіювання, сканування, запис на електронні носії і тому подібне, будь-якої частини посібника без дозволу видавництва заборонено

Рецензенти:

О. В. Новосельцев — член-кореспондент НАН України, доктор технічних наук, професор інституту теплофізики НАН України.

В. І. Сенько — доктор технічних наук, професор Національного технічного університету України «Київський політехнічний інститут».

О. О. Ситник — кандидат технічних наук, професор Черкаського державного технологічного університету.

Рекомендовано Міністерством освіти і науки, молоді та спорту України як навчальний посібник для студентів вищих навчальних закладів (лист Міністерства освіти і науки, молоді та спорту України №1/11-12995 від 08.08.2012 р.)

Матвієнко М. П., Розен В. П.

М 33 Комп'ютерна схемотехніка. Навчальний посібник. — К.: Видавництво Ліра-К, 2020. — 192 с.
ISBN 978–966–2609–14–1

У навчальному посібнику викладено основні поняття комп'ютерної схемотехніки і методи побудови різноманітних комп'ютерних схем та схем автоматики і управління. Теоретичний і практичний матеріал проілюстровано великою кількістю вправ та задач для набуття читачем практичного досвіду.

Навчальний посібник призначено для студентів, аспірантів і спеціалістів, які застосовують відповідні комп'ютерні методи для побудови схем обчислювальної техніки та автоматики, а також окремі розділи посібника можуть бути використані студентами технічних навчальних закладів та коледжів.

**ББК 32.973
УДК 517.1**

ISBN 978–966–2609–14–1

© Матвієнко М. П., Розен В. П. 2012
© «Видавництво Ліра-К», 2012



Передмова	6
Розділ 1. ОСНОВИ ТЕОРІЇ КОМП'ЮТЕРНОЇ СХЕМОТЕХНІКИ	8
1.1. Інформаційні основи комп'ютерної схемотехніки	8
1.2. Арифметичні основи комп'ютерної схемотехніки	11
1.3. Логічні основи комп'ютерної схемотехніки	27
1.4. Автоматні основи комп'ютерної схемотехніки	41
<i>Контрольні запитання</i>	<i>50</i>
<i>Задачі для самостійного розв'язування</i>	<i>51</i>
<i>Коментарі</i>	<i>53</i>
Розділ 2. ОСНОВИ ПОБУДОВИ ЛОГІЧНИХ ЕЛЕМЕНТІВ КОМП'ЮТЕРНОЇ СХЕМОТЕХНІКИ	54
2.1. Діодні логічні елементи	54
2.2. Транзисторні логічні елементи	56
2.3. Діодно-транзисторні логічні елементи	58
2.4. Транзисторно-транзисторні логічні елементи	60
2.5. Логічні елементи на МОН-транзисторах	64
2.6. Функціональні позначення логічних елементів комп'ютерної схемотехніки	68
<i>Контрольні запитання</i>	<i>71</i>
<i>Задачі для самостійного розв'язування</i>	<i>71</i>
<i>Коментарі</i>	<i>71</i>
Розділ 3. СХЕМОТЕХНІКА ПОБУДОВИ КОМБІНАЦІЙНИХ ПРИСТРОЇВ	72
3.1. Схемотехніка побудови дешифраторів і шифраторів	72
3.2. Схемотехніка побудови мультиплексорів і демультиплексорів	77

3.3. Схемотехніка побудови суматорів	81
3.4. Схемотехніка побудови компараторів	86

<i>Контрольні запитання</i>	89
<i>Коментарі</i>	89

Розділ 4. СХЕМОТЕХНІКА ПОБУДОВИ КОМБІНАЦІЙНИХ ПРИБОРІВ НА ПРОГРАМОВАНИХ ЛОГІЧНИХ, МАТРИЦЯХ 90

4.1. Призначення і ділянки застосування	90
4.2. Принципи побудови базової програмуємої логічної матриці	91
4.3. Рекомендації з програмування базової логічної матриці	95
4.4. Програмування базової логічної матриці	95
4.5. Схемотехніка побудови комбінаційних пристроїв на програмованих логічних матрицях	99

<i>Контрольні запитання</i>	102
<i>Коментарі</i>	103

Розділ 5. СХЕМОТЕХНІКА ПОБУДОВИ ТИПОВИХ ПРИБОРІВ ІЗ ПАМ'ЯТТЮ 104

5.1. Схемотехніка побудови RS-тригерів	104
5.2. Схемотехніка побудови D-тригера	109
5.3. Схемотехніка побудови Т-тригера	110
5.4. Схемотехніка побудови JK-тригера	113
5.5. Схемотехніка побудови лічильників	116
5.6. Схемотехніка побудови регістрів	121

<i>Контрольні запитання</i>	123
<i>Коментарі</i>	124

Розділ 6. СХЕМОТЕХНІКА ПОБУДОВИ ІНТЕГРОВАНИХ СИСТЕМ ЕЛЕМЕНТІВ 125

6.1. Схемотехніка побудови одновихідних комбінаційних пристроїв на логічних елементах	125
6.2. Схемотехніка побудови одновихідних комбінаційних пристроїв на мультиплексорах	129
6.3. Схемотехніка побудови багатовихідних комбінаційних пристроїв на логічних елементах	132
6.4. Схемотехніка побудови багатовихідних комбінаційних пристроїв на дешифраторах	136

6.5. Схемотехніка побудови часових логічних пристроїв	138
6.6. Схемотехніка побудови рекурентних логічних пристроїв другого роду	141
6.7. Схемотехніка побудови комп'ютерних пристроїв із застосуванням теорії автоматів	143
6.8. Схемотехніка побудови комп'ютерних пристроїв із застосуванням теорії автоматів і програмованих логічних матриць	145
<i>Контрольні запитання</i>	151
<i>Задачі для самостійного розв'язування</i>	152
<i>Коментарі</i>	153

Розділ 7. СХЕМОТЕХНІКА ПОБУДОВИ АНАЛОГОВИХ СХЕМ 154

7.1. Елементи схемотехніки аналогових вузлів	154
7.2. Основні визначення та характеристики операційних підсилювачів	155
7.3. Схемотехніка інвертуючого підсилювача	159
7.4. Схемотехніка неінвертуючого підсилювача	161
7.5. Схемотехніка побудови суматорів на операційних підсилювачах	162
7.6. Схемотехніка побудови інтеграторів, диференціаторів і компараторів на операційних підсилювачах	164
<i>Контрольні запитання</i>	167
<i>Коментарі</i>	167

Розділ 8. СХЕМОТЕХНІКА ПОБУДОВИ ОБСЛУГОВУЮЧИХ ЕЛЕМЕНТІВ 168

8.1. Схемотехніка перетворювачів рівнів	168
8.2. Схемотехніка генераторів сигналів прямокутної форми	172
8.3. Схемотехніка одновібраторів	176
8.4. Схемотехніка таймерів	180
<i>Контрольні запитання</i>	186
<i>Задачі для самостійного розв'язування</i>	187
<i>Коментарі</i>	187
<i>Література</i>	188



ПЕРЕДМОВА

В навчальному посібнику викладені інформаційні, арифметичні, логічні, автоматні та схемотехнічні основи комп'ютерної схемотехніки. Розглянуті елементи, схеми й типові функціональні вузли комп'ютерів згідно з вимогами навчальних програм відповідно до українських стандартів.

Навчальний посібник ознайомлює читача з теоретичними методами аналізу й синтезу різних пристроїв комп'ютерної схемотехніки, а на їх основі і з логікою побудови комп'ютерних схем. Значна увага приділяється правильному застосуванню точних визначень позначень, і алгоритмам побудови різноманітних комп'ютерних схем.

Частина питань із розглянутих у посібнику недостатньо висвітлена в сучасній навчальній літературі. Зокрема, це стосується дослідження методів структурного синтезу комп'ютерних схем та логіки побудови комп'ютерних схемотехнічних пристроїв.

Усі визначення супроводжено ілюстрованими прикладами. Кожний параграф або розділ закінчується набором ретельно підібраних контрольних запитань та задач для самостійного розв'язування. Практичний матеріал є значним за обсягом і становить близько 30% загального матеріалу посібника. В ньому містяться приклади з рішеннями, контрольні запитання для закріплення і поглиблення розуміння теоретичних положень, задачі для самостійного розв'язування, задля їх практичного застосування, що дає можливість поліпшити організацію самостійної роботи студентів.

У навчальному посібнику зібрано як загальновідомий матеріал, так і розроблений останніми роками, а також подано частково і новий матеріал. Навчальний посібник розрахований, у першу чергу, на студентів, а також читачів, які бажають вивчити інформаційні, арифметичні, логічні, автоматні та схемотехнічні основи комп'ютерної схемотехніки і на їх основі будувати комп'ютерні схеми та схеми автоматики і управління. Від читача вимагається знання в

обсязі середньої школи, а всі подальші знання здобуваються в процесі роботи з книгою та запропонованими в ній вправами, що надає навчальному посібнику привабливого вигляду для практичних застосувань.

Структура книги, що складається із 8 розділів, їх перелік та наповнення легко прочитуються із змісту. Вибір та викладання розділів комп'ютерної схемотехніки виконано, враховуючи вимоги фундаментальної освіти з комп'ютерних дисциплін.

Навчальний посібник призначено для студентів спеціальностей вищих навчальних закладів III–IV рівнів акредитації базових напрямів «Комп'ютерна інженерія», «Комп'ютерні системи, автоматика і управління», «Комп'ютерні науки». Окремі розділи також можуть бути використані студентами відповідних середніх технічних навчальних закладів та коледжів.



Розділ 1

ОСНОВИ ТЕОРІЇ КОМП'ЮТЕРНОЇ СХЕМОТЕХНІКИ

1.1. Інформаційні основи комп'ютерної схемотехніки

Інформація та етапи її представлення

Інформація є відтзеркаленням реального світу і єдиним незаперечним ресурсом життєзабезпечення. Більш того, кількість інформації кожен рік подвоюється, а її обробка майже стовідсотково виконується комп'ютерами. Кожен інформаційний процес включає в себе наступні етапи:

1. Збір інформації і представлення її для вводу в комп'ютер;
2. Передачу інформації від джерела до приймача;
3. Збереження інформації в часі;
4. Обробку інформації в часі;
5. Представлення інформації споживачу.

На всіх, п'яти етапах, як правило, використовують засоби комп'ютерної схемотехніки. До будь-якої інформації завжди застосовуються наступні вимоги:

- а) Інформація завжди повинна бути коректною;
- б) Інформація повинна бути як корисною, так і актуальною;
- в) Інформація повинна бути точною, достовірною і стійкою до будь-яких проявів;
- г) Інформація повинна бути повною для прийняття правильного рішення.

Джерелом та приймачем інформації можуть бути як люди, так і технічні засоби, наприклад: комп'ютери, датчики, індикатори т. ін. Інформацію для передачі від джерела до приймача перетворюють в сигнали, які потім використовують для її передачі. В теорії інформації вважають, що сигнал — це матеріально енергетичне втілення повідомлення. Тобто, за рахунок сукупності сигналів можливо представити будь-яке складне повідомлення. Сигнал може перетворюватися без

зміни змісту інформації із однієї фізичної величини в іншу, більш зручну для передачі і обробки в комп'ютерних пристроях.

Зміна параметрів фізичної величини сигналу за законом повідомлення, що передається, називають **модуляцією**, а змінні параметри — **інформативними**. При передачі сигналів від джерела інформації до приймача фізичні величини і способи їх модуляції можуть багатократно змінюватися, але зміст повідомлення залишається сталим, оскільки воно визначається тільки законом модуляції.

Для інформаційного обміну між джерелом і приймачем використовують **символи, алфавіти та слова**. Під символом розуміють елементарну одиницю повідомлення, яку називають бітом. Якщо число різних символів обмежено, то їх сукупність називають алфавітом, наприклад, букви латинського алфавіту або двійкові символи «0» і «1» у комп'ютерних технічних пристроях. Слова — це групи символів, із яких будують фрази і вирази.

Загалом спосіб формалізованого опису різних сигналів, а відповідно і повідомлень називають **представленням інформації**. В теорії інформації розглядають не фізичне, а математичне представлення сигналів, тобто їх опис за допомогою різних функцій. Найбільш розповсюдженим способом представлення сигналів є **часовий**, в якому розрізняють наступні різновиди сигналів, що описуються часовою функцією: **неперервний, дискретно-неперервний; дискретні**. Спільне застосовування дискретизації і квантування дозволяє перетворити неперервну функцію в суто дискретну. Тобто, неперервний сигнал можна передавати окремими миттєвими значеннями, які відраховуються через кінцевий інтервал часу. За цими значеннями комп'ютер повністю відтворює первинний неперервний сигнал.

В комп'ютерній схемотехніці використовують розрядно-цифрове кодування, в якому первинний сигнал представляють групою символів, які відображають значення цифр «0» і «1» двійкової системи числення електричними сигналами, наприклад, імпульсами. Належність імпульсу відповідає «1», а його відсутність — «0». Розряди двійкового числа характеризуються вагою, кратною степені двійки — 0, 1, 2, 4, ...

Цифровий код передається послідовно в часі за допомогою одного каналу передачі або одночасно, використовуючи багатоканальний спосіб каналу передачі.

На практиці послідовний код використовують при передачі інформації на великі відстані, наприклад, між комп'ютерами, а паралельний — при передачі інформації на невеликі відстані, наприклад, всередині комп'ютера.

Підрахунок кількості інформації

Для підрахунку кількості переданої чи прийнятої інформації в теорії інформації встановлені різні інформаційні міри. Найбільш поширена є адитивна міра. Суть її полягає в наступному. Нехай N — кількість рівноймовірних повідомлень, n — їх довжина, q — число букв алфавіту, що використовується для передачі інформації. Тоді кількість можливих повідомлень довжини n дорівнює числу розміщень із повторенням. Цю міру наділяють властивістю адитивності, щоб вона була пропорційна довжині повідомлення і дозволила додавати кількість інформації ряду джерел. Для цього вчений Хартлі запропонував логарифмічну функцію, як міру кількості інформації

$$I = \log N = n \cdot \log q. \quad (1.1.1)$$

Якщо для алфавіту використовують двійкові чисел «0» і «1», то за основу логарифму приймають $q = 2$, в результаті чого $I = n \cdot \log_2 2 = n$. При довжині $n = 1$, отримаємо $I = 1$, цю кількість інформації називають **бітом**. Передача повідомлення довжиною $n = 1$ еквівалентна вибору одного із двох можливих рівноймовірних повідомлень — одне із яких дорівнює «1», а друге — «0». В загальному випадку повідомлення з'являються з різною ймовірністю. Нехай повідомлення утворюється послідовною передачею букв деякого алфавіту $x_1, \dots, x_i, \dots, x_n$ із ймовірністю появи кожної букви $p(x_1) = p_1, \dots, p(x_n) = p_n$, при цьому повинна виконуватись умова $p_1 + \dots + p_i + \dots + p_n = 1$. Тоді, за Шенноном, кількість інформації, яка є в повідомленні x_i , розраховується за формулою

$$I(x_i) = \log \frac{1}{p_i}. \quad (1.1.2)$$

Для абсолютно достовірних повідомлень $p_i = 1$; кількість інформації $I(x_i) = 0$. При зменшенні значення p_i кількість інформації збільшується. Якщо в ансамблі повідомлень усі букви алфавіту $x_1, \dots, x_i, \dots, x_n$ — рівномірні, то $p_1 = p_2 = \dots = p_n = \frac{1}{n}$ є статистично

незалежні. Тоді кількість інформації в повідомленні довжиною n букв з урахуванням 1.1.2 буде дорівнювати

$$I = \sum_{i=1}^n I(x_i) = \sum_{i=1}^n \log \frac{1}{p_i} = \log \frac{1}{p_1} + \dots + \log \frac{1}{p_n} = n \cdot \log q,$$

що співпадає з мірою Хартлі у відповідності з виразом 1.1.1.

В комп'ютері найменшою можливою одиницею кількості інформації є біт. Для зручності використання введені також одиниці кількості інформації, які перевищують біт. Так, двійкове слово із восьми символів має 1 байт інформації, 1024 байт складає кілобайт (Кбайт), 1024 Кбайт — мегабайт і 1024 Мбайт — гігабайт (Гбайт).

1.2. Арифметичні основи комп'ютерної схемотехніки Принципи побудови систем числення

Означення 1.2.1. Системою числення називають систему відображення будь-яких чисел за допомогою обмеженого числа знаків.

В залежності від способів відображення чисел цифрами, системи числення діляться на позиційні і непозиційні.

Означення 1.2.2. Непозиційною системою числення називають систему, в якій кількісне значення кожної цифри не залежить від місця у відображенні числа, а визначається лише самим символом числа. Так, наприклад, число 30 десяткової системи числення в римській непозиційній системі позначають як число XXX, яке має у всіх розрядах один і той же символ X, що означає 10 одиниць незалежно від його позиції у відображенні числа.

Означення 1.2.3. Позиційною системою числення називають систему, в якій кількісне значення кожної цифри залежить від її місця у відображенні числа. Наприклад, число 575, представлене в десятковій системі числення, має в найстаршому і наймолодшому розряді цифру 5. Цифра 5 у старшому розряді має вагу в 100 раз більшу, ніж у молодшому розряді.

В позиційній системі числення будь-яке число, яке має відображення

$$A_n = \pm \alpha_1 \cdot \alpha_2 \cdot \alpha_3 \dots \alpha_{i-1} \cdot \alpha_i$$

може бути представлено у вигляді такої суми

$$\alpha_1 \cdot \alpha_2 \cdot \alpha_3 \dots \alpha_{k-1} \cdot \alpha_k = \alpha_1 \cdot q^{t-1} + \alpha_2 \cdot q^{t-2} + \dots + \alpha_{k-1} \cdot q^{t-k+1} + \alpha_k \cdot q^{t-k}, \quad (1.2.1)$$

де k — кінцева кількість розрядів у відображенні числа; α_i — цифра i -го розряду; q — основа системи числення; t — фіксоване число, яке визначає розміщення коми; i — порядковий номер розряду; q^{t-i} — вага i -го розряду. Цифри α_i повинні задовільняти нерівності $0 \leq \alpha_i \leq q-1$.

Означення 1.2.4. Основою системи числення q називають кількість символів, які використовують для відображення числа в даній позиційній системі числення.

За основу системи числення q мають будь-яке число, яке задовольняє умові $q \geq 2$. У двійковій системі числення $q = 2$, а для зображення чисел використовують символи (1,0), у вісімковій $q = 8$, і для зображення чисел використовують символи (0, 1, 2, 3, 4, 5, 6, 7), а у шістнадцятковій $q = 16$ і для зображення чисел використовують символи (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F), де $A = 10$; $B = 11$; $C = 12$; $D = 13$; $E = 14$; $F = 15$.

Двійкова система числення є основною системою числення, в якій виконують арифметичні і логічні операції в комп'ютерах, тому що для її технічної реалізації широко застосовування знайшли дво-позиційні електронні елементи.

Переведення чисел із однієї системи числення в іншу

Суттєве значення при виконанні арифметичних і логічних операцій в комп'ютерах має питання переведення чисел із однієї системи числення в іншу. Для переведення цілих чисел найчастіше використовують алгоритм ділення заданого числа на основу числа, до системи якої його переводять. Даний алгоритм переведення чисел із системи числення з основою q є універсальним і найбільш широко застосовується на практиці. Він містить такі кроки.

1. Розділити число, яке переводять, у системі числення з основою q на основу p за правилом системи числення з основою q .
2. Перевірити, чи не дорівнює частка нулю. Якщо не дорівнює, то прийняти її за нове число й повернутися до кроку 1.
3. Якщо частка дорівнює нулю, то виписати всі отримані залишки від ділення в порядку, зворотному їх отриманню.

4. Отриманий запис є записом числа в системі числення з основою p .

Для переведення дробових чисел використовують алгоритм множення даного числа на основу числа, до системи якої його переводять. Даний алгоритм містить такі кроки.

1. Помножити дробове число, з основою q на основу системи числення p , у яку його переводять.
2. В кожному добутку виділити цілі частини, якщо такі є.
3. Виділені цілі частини добутку і є цифрами дробової частини числа.

Реалізація даних алгоритмів при переведенні цілих чисел і дробових чисел із точністю 10^{-4} з десяткової системи числення у двійкову, вісімкову та шістнадцяткову приведено в табл. 1.2.1.

Таблиця 1.2.1

Тип перетворення	Цілі числа	Дробові числа
Десяткове на двійкове	$125 : 2 = \text{остача } 1$ $62 : 2 = \text{остача } 0$ $31 : 2 = \text{остача } 1$ $15 : 2 = \text{остача } 1$ $7 : 2 = \text{остача } 1$ $3 : 2 = \text{остача } 1$ $1 : 2 = \text{остача } 1 \uparrow$ $125_{(10)} = 1111101_{(2)}$	$0,45 \cdot 2 = \text{переноситься } 0 \downarrow$ $0,90 \cdot 2 = \text{переноситься } 1 (1,80)$ $0,80 \cdot 2 = \text{переноситься } 1 (1,60)$ $0,60 \cdot 2 = \text{переноситься } 1 (1,20)$ Переривання $0,45_{(10)} = 0,0111_{(2)}$
Десяткове на вісімкове	$125 : 8 = \text{остача } 5$ $15 : 8 = \text{остача } 7 \uparrow$ $125_{(10)} = 75_{(8)}$	$0,45 \cdot 8 = \text{переноситься } 3 (3,60) \downarrow$ $0,60 \cdot 8 = \text{переноситься } 4 (4,80)$ $0,80 \cdot 8 = \text{переноситься } 6 (6,40)$ $0,40 \cdot 8 = \text{переноситься } 3 (3,20)$ Переривання $0,45_{(10)} = 0,3463_{(8)}$
Десяткове на шістнадцяткове	$125 : 16 = 7$ $13 : 16 = 13 \uparrow$ $125_{(10)} = D7_{(16)}$	$0,45 \cdot 16 = \text{переноситься } 7 (7,20) \downarrow$ $0,20 \cdot 16 = \text{переноситься } 3 (3,20)$ $0,20 \cdot 16 = \text{переноситься } 3 (3,20)$ $0,20 \cdot 16 = \text{переноситься } 3 (3,20)$ Переривання $0,45_{(10)} = 0,7333_{(16)}$

В табл. 1.2.1. стрілкою показано напрям запису переведених чисел із десяткової в двійкову, вісімкову та шістнадцяткову системи числення.

Переведення цілого числа двійкової системи числення в десяткову відбувається згідно з формулою

$$a_n \cdot 2^n + a_{n-1} \cdot 2^{n-1} + \dots + a_1 \cdot 2^1 + a_0 \cdot 2^0 = A_1, \quad (1.2.2)$$

а дробового — згідно з формулою

$$a_1 \cdot 2^{-1} + a_2 \cdot 2^{-2} + \dots + a_{n-1} \cdot 2^{-(n-1)} + a_n \cdot 2^{-n} = A_2, \quad (1.2.3)$$

де $a_0, a_1, a_2, \dots, a_{n-1}, a_n$ — цифри двійкового числа, які приймають значення «0» або «1»; A_1 — ціле десяткове число; A_2 — дробове десяткове число.

Переведення цілого числа вісімкової системи числення в десяткову відбувається згідно з формулою

$$b_n \cdot 8^n + b_{n-1} \cdot 8^{n-1} + \dots + b_1 \cdot 8^1 + b_0 \cdot 8^0 = B_1, \quad (1.2.4)$$

а дробового — згідно з формулою

$$b_1 \cdot 8^{-1} + b_2 \cdot 8^{-2} + \dots + b_{n-1} \cdot 8^{-(n-1)} + b_n \cdot 8^{-n} = B_2, \quad (1.2.5)$$

де $b_0, b_1, b_2, \dots, b_{n-1}, b_n$ — цифри вісімкового числа; B_1 — ціле десяткове число; B_2 — дробове десяткове число.

Переведення цілого числа шістнадцяткової системи числення в десяткову відбувається за формулою

$$c_n \cdot 16^n + c_{n-1} \cdot 16^{n-1} + \dots + c_1 \cdot 16^1 + c_0 \cdot 16^0 = C_1, \quad (1.2.6)$$

а дробового — за формулою

$$c_1 \cdot 16^{-1} + c_2 \cdot 16^{-2} + \dots + c_{n-1} \cdot 16^{-(n-1)} + c_n \cdot 16^{-n} = C_2, \quad (1.2.7)$$

де $c_0, c_1, c_2, \dots, c_{n-1}, c_n$ — цифри шістнадцяткового числа; C_1 — ціле десяткове число; C_2 — дробове десяткове число.

Зробити перевірку правильності переведення цілої і дробової частин десяткових чисел у двійкове, вісімкове та шістнадцяткове число, яке виконано в табл. 1.2.1., використовуючи формули 1.2.2, ..., 1.2.7, пропонується читачеві зробити це самостійно.

При переведенні змішаних чисел із десяткової системи числення в інші необхідно виконати переведення цілої і дробової частин окремо згідно з їх алгоритмами, а отримані результати об'єднати.

Дуже часто при переведенні десяткових чисел у двійкові використовують проміжне переведення їх у вісімкову або шістнадцяткову системи числення з подальшим записом кожної їх цифри у вигляді трьох або чотирьохзначного двійкового числа.

Приклад 1.2.1. Перевести число $257,36_{(8)}$ вісімкової системи числення у двійкову.

Розв'язання. Записуючи кожну цифру вісімкового числа у вигляді триад двійкового числа, отримаємо $257,36_{(8)} = 010\ 101\ 111,011\ 110_{(2)}$.

Приклад 1.2.2. Перевести число $A7FD, B5C_{(16)}$ шістнадцяткової системи числення у двійкову.

Розв'язання. Записуючи кожну цифру шістнадцяткової системи числення у вигляді тетрад двійкового числа, отримаємо

$$A7FD, B5C_{(16)} = 1010\ 0111\ 1111\ 1101, 1011\ 0101\ 1100_{(2)}$$

Переведення двійкових чисел у вісімкову систему числення відбувається слідуючим чином. Задане двійкове число розподіляють на триади справа наліво — для цілих чисел і зліва направо — для дробових чисел, а для змішаних — від коми, як вліво так і вправо.

Переведення двійкових чисел у шістнадцяткову систему числення відбувається аналогічно, як і для вісімкової системи числення, але розподіл двійкового числа відбувається не на триади, а на тетради.

Переведення двійкових чисел у двійково-десяткові відбувається шляхом поділу двійкового числа на тетради справа наліво для цілих чисел, а для змішаних від коми як уліво, так і вправо.

Арифметичні операції над двійковими числами

Є чотири арифметичні операції в двійковій системі числення (додавання, віднімання, множення і ділення).

Додавання. При додаванні двійкових чисел необхідно користуватись наступними правилами:

$$a) 0 + 0 = 0; \quad б) 0 + 1 = 1; \quad в) 1 + 0 = 1; \quad г) 1 + 1 = 10. \quad (1.2.8)$$

Двозначна сума додавання в 1.2.8 з означає, що при додаванні двійкових цифр, які дорівнюють 1, виникає перенос 1 до сусіднього старшого розряду. Цей перенос повинен бути доданий до суми цифр, які утворюються в сусідньому розряді зліва.

Приклад 1.2.3. Виконати додавання двох двійкових чисел $A_1 = 1101101$ і $A_2 = 1001111$.

Розв'язання. У відповідності з 1.2.8, виконуємо порозрядне додавання двійкових чисел A_1 і A_2

$$\begin{array}{r} 1101101 \quad - A_1 \\ 1001111 \quad - A_2 \\ \hline 0100010 \quad - \text{порозрядна сума без урахування переносів} \\ + \\ 1 \quad 1111 \quad - \text{переноси} \\ \hline 10111100 \quad - A_1 + A_2 \end{array}$$

Безпосередньо під двома доданками записаний результат порозрядного додавання без урахування переносів. У тих розрядах, в яких обидва доданки дорівнюють «1», порозрядна сума дорівнює «0», і в цих розрядах утворилося перенесення в сусідній старший розряд, що відмічений у наступній стрічці «1». В результаті додавання стрічки порозрядних сум зі стрічкою переносів, отримаємо кінцеву суму.

Віднімання. При відніманні двійкових чисел необхідно користуватись такими правилами:

$$\text{а) } 0 - 0 = 0; \text{ б) } 1 - 0 = 1; \text{ в) } 1 - 1 = 0; \text{ г) } 10 - 1 = 1. \quad (1.2.9)$$

Якщо при порозрядному відніманні необхідно відняти із нуля одиницю, то береться позичка в сусідньому розряді, тобто одиниця старшого розряду представляється як дві одиниці даного розряду. Віднімання в цьому випадку виконується у відповідності з 1.2.9 з. Якщо в сусідньому старшому розряді або декількох старших розрядах містяться нулі, то позичка береться у найближчому старшому розряді з одиниці. Ця одиниця представляється у вигляді суми числа, яке складається із одиниць у всіх проміжних розрядах, в яких містились нулі і двох одиниць у даному розряді. Віднімання у даному розряді виконується у відповідності з 1.2.9 з, а у всіх проміжних розрядах — згідно з 1.2.9 а, ..., 1.2.9 в.

Приклад 1.2.4. Виконати віднімання двох двійкових чисел $A_1 = 11000011$ і $A_2 = 10100110$.

Розв'язання. У відповідності з 1.2.9, виконаємо порозрядне віднімання двійкових чисел A_1 і A_2

$$\begin{array}{r} 11000011 \quad - A_1 \\ - 10100110 \quad - A_2 \\ \hline 00011101 \quad - A_1 - A_2 \end{array}$$

У двох молодших розрядах не було необхідності робити позичку. Для третього розряду позичка зроблена із сьомого розряду (найближча одиниця). В проміжних розрядах віднімання відбувається із одиниці.

Множення. Множення двох двійкових чисел виконується так само, як і множення двох десяткових. Тобто, множене послідовно множиться на кожен цифру множника, розпочинаючи з молодшої або зі старшої. Для врахування ваги відповідної цифри множника результат рухається або вліво, якщо множення відбувається, починаючи з молодшого розряду множника, або вправо, якщо множення відбувається, починаючи із старшого розряду множника. При цьому рух відбувається на таке число розрядів, на яке відповідний розряд множника зсунутий відносно молодшого або старшого його розряду. Отримані в результаті множення і зсуву часткові добутки після додавання дають повний результат множення.

Приклад 1.2.5. Виконати множення двох двійкових чисел $A_1 = 10101$ і $A_2 = 1011$ зсувом вліво і вправо.

Розв'язання. У відповідності з викладеними вище правилами множення, воно матиме наступний вигляд для зсуву

Вліво	Вправо
10101	10101
×	×
1011	1011
10101	10101
10101	10101
10101	10101
11100111	11100111

Якщо множене або множник, або разом, мають цілу і дробову частини, то множення таких чисел відбувається як множення

З формули 1.2.11 і прикладу випливає, що додатковий код додатного числа співпадає з зображенням додатного числа прямого коду, а для зображення від'ємного числа в додатковому коді необхідно у знаковому розряді записати одиницю, а у всіх числових розрядах нулі замінити на одиниці, а одиниці — нулями і до отриманого результату додати одиницю до молодшого розряду.

В додатковому коді від'ємний нуль відсутній.

Обернений код. Формула для зображення оберненого коду двійкового числа A має такий вигляд:

$$A_o = \begin{cases} A, & \text{якщо } A \geq 0, \\ 10 + A - 10^n, & \text{якщо } A < 0. \end{cases} \quad (1.2.12)$$

Приклад 1.2.9. Записати двійкові числа $A_1 = +0,10110$ і $A_2 = -0,10110$ у оберненому коді.

Розв'язання. У відповідності з формулою 1.2.12, дані двійкові числа матимуть такий вигляд: $A_{1o} = 0,10110$, $A_{2o} = 10 - 0,10110 - 0,00001 = 1,01001$.

Із формули 1.2.12 і прикладу випливає, що обернений код додатного числа співпадає із зображенням додатного числа прямого коду, а для зображення від'ємного числа в оберненому коді необхідно у знаковому розряді записати одиницю, і у всіх числових розрядах нулі замінити на одиниці, а одиниці — нулями.

Модифіковані коди. Ці коди використовують для виявлення переповнення розрядної сітки, яке може статися при додаванні двійкових чисел. Модифіковані коди відрізняються від простих машинних кодів тим, що на відображення знака в них відводиться два розряди. Плюс відображається двома нулями, а мінус — двома одиницями.

Переведення двійкових чисел у модифіковані прямі, додаткові та обернені коди відбувається за правилами, наведеними в формулах 1.2.10, 1.2.11 і 1.2.12.

Приклад 1.2.10. Записати двійкові числа $A_1 = +0,11010$ і $A_2 = -0,11010$ в прямому, додатковому і оберненому модифікованих кодах.

Розв'язання. У відповідності з визначенням модифікованого коду, а також використовуючи формули 1.2.10, 1.2.11 і 1.2.12, дані коди матимуть такий вигляд:

$$\begin{aligned} A_{1n}^m &= 00,11010, & A_{1o}^m &= 00,11010, & A_{1o}^m &= 00,11010, \\ A_{2n}^m &= 11,11010, & A_{2o}^m &= 11,00110, & A_{2o}^m &= 11,00101. \end{aligned}$$

Додавання і віднімання двійкових чисел

Додавання чисел у модифікованому додатковому коді з фіксованою комою відбувається за правилами двійкової арифметики, наведеної вище. Одиниця переносу, яка виникає у старшому знаковому розряді суми, відкидається. Знаковим розрядом числа є другий зліва від коми розряд, а перший використовується для аналізу переповнення розрядної сітки.

Приклад 1.2.11. В модифікованому додатковому коді додати двійкові числа A_1 і A_2 за умови:

$$\text{а) } A_1 > 0; A_2 > 0; A_1 + A_2 > 0; A_1 = +0,1101; A_2 = +0,0001.$$

$$A_{1o}^m = 00,1101$$

+

$$A_{2o}^m = 00,0001$$

$$A_1 + A_2 = 00,1110; \quad (A_1 + A_2)_n^m = 00,1110.$$

$$\text{б) } A_1 > 0; A_1 < 0; A_1 + A_2 > 0; A_1 = +0,1101; A_2 = -0,0001.$$

$$A_{1o}^m = 00,1101$$

+

$$A_{2o}^m = 11,1111$$

$$(A_1 + A_2)_n^m = 00,1100; \quad (A_1 + A_2)_n^m = 00,1100.$$

Одиниця переносу зі старшого знакового розряду не враховується.

$$\text{в) } A_1 < 0; A_2 > 0; A_1 + A_2 < 0; A_1 = -0,1101; A_2 = +0,0001.$$

$$A_{1o}^m = 11,0011$$

+

$$A_{2o}^m = 00,0001$$

$$(A_1 + A_2)_n^m = 11,0100; \quad (A_1 + A_2)_n^m = 11,1100.$$

$$\begin{array}{r} r) A_1 < 0; A_2 < 0; A_1 + A_2 < 0; A_1 = -0,1101; A_2 = -0,0001. \\ A_{10}^m = 11,0011 \\ + \\ A_{20}^m = 11,1111 \\ \hline (A_1 + A_2)_0^m = 11,0010; \quad (A_1 + A_2)_n^m = 11,1110. \end{array}$$

Одиниця переносу зі старшого знакового розряду не враховується.

Додавання чисел у модифікованому оберненому коді з фіксованою комою виконується так, як і в додатковому коді. Різниця полягає тільки в тому, що одиницю переносу зі старшого знакового розряду (якщо вона є) необхідно додати до молодшого розряду суми.

Приклад 1.2.12. В модифікованому оберненому коді додати двійкові числа A_1 і A_2 за умови:

$$\begin{array}{r} A_1^m = 00,1101 \\ + \\ A_2^m = 00,0001 \\ \hline (A_1 + A_2)_o^m = 00,1110; \quad (A_1 + A_2)_n^m = 00,1110. \end{array}$$

$$\begin{aligned} & \text{б) } A_1 > 0; A_2 < 0; A_1 + A_2 > 0; A_1 = +0,1101; A_2 = -0,0001 \\ & \quad A_{1o}^m = 00,1101 \\ & \quad + \\ & \quad A_{2o}^m = 11,1110 \\ & \quad \hline & (A_1 + A_2)_o^m = 100,1011; \\ & \qquad \searrow +1 \text{ циклічний перенос} \\ & (A_1 + A_2)_o^m = 00,1100, (A_1 + A_2)_n^m = 00,1100. \end{aligned}$$

$$\begin{array}{r} \text{B) } A_1 < 0; A_2 > 0; A_1 + A_2 < 0; A_1 = -0,1101; A_2 = +0,0001. \\ A_{1o}^m = 11,0010 \\ + \\ A_{2o}^m = 00,0001 \\ \hline (A_1 + A_2)_o^m = 11,0011; \quad (A_1 + A_2)_n^m = 11,1100. \end{array}$$

$$\begin{array}{r} \text{г) } A_1 < 0; A_2 < 0; A_1 + A_2 < 0; A_1 = -0,1101; A_2 = -0,0001. \\ A_{1o}^m = 11,0010 \\ + \\ A_{2o}^m = 11,1110 \\ \hline (A_1 + A_2)_o^m = 111,0000; \\ \quad \quad \quad \downarrow \rightarrow +1 \text{ циклічний перенос} \\ (A_1 + A_2)_o^m = 11,0001, \quad (A_1 + A_2)_n^m = 11,1110. \end{array}$$

Переповнення розрядної сітки при додаванні в простих кодах.

При додаванні двох двійкових чисел, абсолютною величиною менші одиниці, код суми за абсолютною величиною може бути більшим за одиницю або дорівнювати їй. У цьому випадку буде переповнення розрядної сітки, що приведе до спотворення результату додавання.

Приклад 1.2.13. У додатковому коді додати двійкові числа A_1 і A_2 за умови:

а) $A_1 > 0; A_2 > 0; A_1 + A_2 > 0; A_1 = +0,1101; A_2 = +0,0111.$
 $A_{10}^m = 0,1101$
 $+$
 $A_{20}^m = 0,0111$

 $(A_1 + A_2)_0 = 1,0100; (A_1 + A_2)_n^m = 1,1100.$

б) $A_1 < 0; A_2 < 0; A_1 + A_2 < 0; A_1 = -0,1101; A_2 = -0,0111.$
 $A_{1d} = 1,0011$
 $+$
 $A_{2d} = 1,0011$

 $(A_1 + A_2)_0 = 10,1100; (A_1 + A_2)_n^m = 0,1100.$

Виходячи з прикладу 1.2.13, додавання в додатковому двійковому простому коді привело до повного спотворення результату як за знаком, так і за значенням числа через переповнення розрядної сітки. В першому випадку (1.2.13 а) переповнення розрядної сітки відбулося шляхом переносу в знаковий розряд за відсутності переносу з розряду знака, а в другому випадку (1.2.13 б) переповнення відбулося в знаковому розряді за відсутності його в розряді числа. Для уникнення таких спотворень і використовують модифіковані машинні коди.

Переповнення розрядної сітки при додаванні в модифікованих машинних кодах виявляють шляхом порівняння знакових розрядів отриманої суми. Дане твердження покажемо на такому прикладі.

Приклад 1.2.14. Виконати додавання двійкових чисел A_1 і A_2 , заданих в умові прикладу 1.2.16, використовуючи модифікований машинний код.

Розв'язання. Використовуючи умови прикладу 1.2.13, а також представлення двійкових чисел у додатковому коді, отримаємо:

$$\begin{array}{r} \text{а)} \quad A_{10}^m = 00,1101 \\ + \\ A_{20}^m = 00,0111 \\ \hline A_{10}^m + A_{20}^m = 01,0100; \end{array}$$

$$\begin{array}{r} \text{б)} \quad A_{10}^m = 11,0011 \\ + \\ A_{20}^m = 11,1001 \\ \hline A_{10}^m + A_{20}^m = 110,1100; \\ \quad \quad \quad \uparrow \text{— зникає} \end{array}$$

Виходячи з прикладу 1.2.14, у знакових розрядах отриманої суми додатних доданків маємо комбінацію «01», а від'ємних — «10», що є ознакою переповнення розрядної сітки.

Додавання і віднімання двійкових чисел із плаваючою комою відбувається за три кроки. На першому кроці необхідно виконати **вирівнювання порядків** чисел. У будь-якому числі з плаваючою комою вага N_i одиниці i -го розряду мантиси визначається не тільки позицією даного розряду, але і порядком p числа, тобто $N_i = 2^{p-i}$, де i — номер позиції, рахуючи вправо від коми.

При додаванні мантис необхідно, щоб вага одиниць однойменних розрядів мантиси чисел була однаковою. Для цього мантиси переміщуються відносно одна одної так, щоб їх порядки стали рівними. При вирівнюванні порядків отримувати мантиси більше одиниці не дозволяється і тоді їх необхідно рухати у бік більшого порядку. Мантиси з меншим порядком рухаються вправо на кількість розрядів, яке дорівнює різниці порядків.

На другому кроці відбувається **додавання мантис** із вирівняними порядками, яке відбувається аналогічно додаванню чисел із фіксованою комою. Для додавання від'ємних мантис використовують додатковий або обернений модифікований двійковий код. Сума мантис є сумою чисел. Порядок суми чисел дорівнює спільному порядку доданків, тобто порядку більшого числа.

На третьому кроці відбувається **нормалізація отриманого результату**. При додаванні мантис за абсолютною величиною менших одиниці можливі випадки переповнення розрядної сітки — порушення нормалізації зліва від коми. Ознакою такого порушення нормалізації є поява в знакових розрядах модифікованого коду суми різних цифр (наприклад, 01,101... або 10,101...). Для встановлення нормалізації мантиса суми рухається на один розряд вліво, а порядок суми збільшується на одиницю. При додаванні мантис можливі випадки отримання результату, меншого 0,1 (двійкове число), тобто порушення нормалізації справа від коми. Ознакою такого порушення є значення в мантисі суми одного або підряд декількох нулів справа від коми. При нормалізації вліво мантиса рухається вправо на таку кількість розрядів, щоб старша цифра мантиси була відмінна від нуля. Порядок суми зменшується на кількість одиниць, рівній числу рухів мантиси при нормалізації. Якщо всі розряди мантиси суми дорівнюють нулю, то нормалізація не відбувається, а порядок суми дорівнює нулю.

Приклад 1.2.15. Використовуючи плаваючу кому, додати два двійкові числа

$$A_1 = +0,10100 \cdot 10^{+101} \text{ і } A_2 = -0,10110 \cdot 10^{+100}.$$

Розв'язання. Розв'язання виконаємо за три кроки. На першому кроці для вирівнювання порядків доданків необхідно із порядку числа A_1 відняти порядок числа A_2 . Віднімання замінюємо додаванням порядків у модифікованому додатковому коді

$$\begin{array}{r} P_{A_1}^m = 00,101 \\ + \\ P_{A_2}^m = 11,100 \\ \hline (P_{A_1} + P_{A_2})^m = 00,001. \end{array}$$

Так як P_{A_1} на одиницю більше P_{A_2} , то рухаємо мантису числа A_1 вправо на один розряд, тобто $M_{A_1}^m = 1,10110 \rightarrow 1,01011$.

На другому кроці додаємо мантиси чисел A_1 і A_2 в модифікованому додатковому коді

$$M_{A_1}^m = 00,10100$$

+

$$M_{A_2}^m = 11,10101$$

$$(M_{A_1} + M_{A_2})_0^m = 00,01001.$$

На третьому кроці нормалізуємо результат $0,01001 \cdot 10^{+101}$. Для цього рухаємо мантису суми на один розряд вліво та зі значення порядку суми віднімаємо одиницю. В результаті цього отримаємо нормалізований результат додавання чисел $A_1 + A_2 = 0,1001 \cdot 10^{+100}$.

Приклад 1.2.16. Використовуючи плаваючу кому, додати два двійкових числа $A_1 = +0,10100 \cdot 10^{+011}$ і $A_2 = +0,11100 \cdot 10^{+111}$.

Розв'язання. Вирівнюємо порядки доданків

$$P_{A_1}^m = 00,011$$

+

$$P_{A_2}^m = 11,111$$

$$(P_{A_1} + P_{A_2})_0^m = 00,010; \quad (P_{A_1} + P_{A_2})_n^m = 00,010.$$

Як і в попередньому прикладі, для віднімання використаємо модифікований код. Так як P_{A_1} на дві одиниці менше P_{A_2} , то рухаємо вправо мантису $M_{A_1}^m = 0,10100 \rightarrow 0,00101$. Тепер додаємо мантиси чисел A_1 і A_2 та отримаємо

$$M_{A_1}^m = 00,00101$$

+

$$M_{A_2}^m = 00,11100$$

$$(M_{A_1} + M_{A_2})_0^m = 01,00001.$$

Нормалізуємо отриманий результат додаванням чисел $A_1 + A_2 = 1,00001 \cdot 10^{+101}$, для чого рухаємо мантису на один розряд вправо і збільшуємо порядок числа на одиницю. В результаті цього отриманий результат матиме такий вигляд: $A_1 + A_2 = 0,100001 \cdot 10^{+110}$.

1.3. Логічні основи комп'ютерної схемотехніки

Основні визначення

Означення 1.3.1. Функцію $f(x_1, x_2, \dots, x_n)$ називають **логічною** (булевою), якщо вона також, як і її змінні $\langle x_1, x_2, \dots, x_n \rangle$, приймає лише два значення 0 і 1, які називають логічними константами.

Означення 1.3.2. Дві логічні функції $f_1(x_1, x_2, \dots, x_n)$ і $f_2(x_1, x_2, \dots, x_n)$ називають **рівними**, якщо вони приймають однакові значення на всіх можливих наборах їх змінних.

Означення 1.3.3. Змінну x_i у логічній функції називають **істотною**, якщо для неї виконується умова $f(x_1, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n) \neq f(x_1, \dots, x_{i-1}, 1, x_{i+1}, \dots, x_n)$.

Означення 1.3.4. Змінну x_i у логічній функції називають **неістотною** (фіктивною), якщо $f(x_1, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n) = f(x_1, \dots, x_{i-1}, 1, x_{i+1}, \dots, x_n)$ на будь-якому наборі не змінює значення функції.

Елементарні логічні функції

До елементарних логічних функцій належать дванадцять функцій. Функцію $f_0(x, y) = 0$ називають **константою нуль**.

Функцію $f_1(x, y) = x \& y$ називають **кон'юнкцією** або **логічним множенням** і позначають символами $\langle \&, \wedge, \cdot \rangle$. Ця функція приймає значення «1» тоді і тільки тоді, коли логічні змінні x і y приймають значення «1» одночасно.

Функцію $f_2(x, y) = x \Delta y$ називають **заборона за y або заперечення імплікації**. Логічна функція заборона за y приймає логічне значення «1» тоді і тільки тоді, коли логічна змінна x приймає значення «1», а логічна змінна y — «0». Для її позначення використовують символи $\langle \Delta, \leftarrow \rangle$.

Функцію $f_3(x, y) = x$ називають **повторенням** першого аргументу.

Функцію $f_4(x, y) = x \oplus y$ називають **виключаючою «АБО»**, **сумою за модулем два** або функцією **нерівнозначності**. Ця функція приймає значення «1» тоді і тільки тоді, коли значення логічної змінної x дорівнює «1», а $y = «0»$ і навпаки. Для її позначення застосовують символи $<\oplus, \neq>$.

Функцію $f_5(x, y) = x \vee y$ називають **диз'юнкцією** або **логічним додаванням**. Ця функція приймає значення «1» тоді і тільки тоді, коли хоча б одна змінна приймає значення «1». Для її позначення застосовують символи $<\vee, +>$.

Функцію $f_6(x, y) = x \downarrow y$ називають **стрілкою Пірса** або **запереченням диз'юнкції**. Ця функція приймає значення «1» тоді і тільки тоді, коли логічні змінні x і y приймають значення «0» одночасно. Для її позначення використовують символ $<\downarrow>$.

Функцію $f_7(x, y) = x \sim y$ називають **еквівалентністю** або **рівнозначністю**. Ця функція приймає значення «1» тоді і тільки тоді, коли логічні змінні x і y набувають значення «0» або «1» одночасно. Для її позначення використовують символи $<\sim, \leftrightarrow, \equiv>$.

Функцію $f_8(x, y) = \bar{x}$ називають **запереченням** або **інверсією** першого елемента. Для її позначення використовують символи $<\bar{}, \neg>$.

Функцію $f_9(x, y) = x \rightarrow y$ називають **імплікацією**. Ця функція приймає значення «0» тоді і тільки тоді, коли логічна змінна x набуває значення «1», а змінна $y = «0»$. Для її позначення використовують символи $<\rightarrow, \Rightarrow>$.

Функцію $f_{10}(x, y) = x | y$ називають функцією **Шеффера**, або **запереченням кон'юнкції**. Ця функція приймає логічне значення «0» тоді і тільки тоді, коли логічні змінні x і y набувають значення «1» одночасно. Для її позначення використовують знак $<|>$.

Функцію $f_{11}(x, y) = 1$ називають **константою «1»**.

Наведені дванадцять функцій, їх позначення, назва і прочитання представлені в таблиці 1.3.1.

Таблиця 1.3.1

Функція	Позначення	Назва	Прочитання
$f_0(x, y)$	0	Константа 0	Константа 0
$f_1(x, y)$	$x \& y = x \wedge y = \bar{x} \cdot y$	Кон'юнкція (логічне «І»)	x і y
$f_2(x, y)$	$x \Delta y$	Заборона або заперечення імплікації	x і не y
$f_3(x, y)$	x	Повторення першого аргументу	Як x
$f_4(x, y)$	$x \oplus y$	Що виключає «АБО» (сума за модулем 2)	x не як y
$f_5(x, y)$	$x \sqcup y = x + y$	Диз'юнкція (логічне «АБО»)	x або y
$f_6(x, y)$	$x \downarrow y$	Стрілка Пірса (заперечення диз'юнкції)	Не x і не y
$f_7(x, y)$	$x \sim y$	Еквівалентність	x як y
$f_8(x, y)$	$\bar{x}$	Заперечення першого елемента	Не x
$f_9(x, y)$	$x \rightarrow y$	Імплікація	Не x або y
$f_{10}(x, y)$	$x y$	Функція Шеффера (заперечення кон'юнкції)	Не x або не y
$f_{11}(x, y)$	1	Константа 1	Константа 1

Основні закони алгебри логіки

В алгебрі логіки необхідно виділити наступні основні закони: **комутативність; асоціативність; дистрибутивність; закони для заперечення; нулі та одиниці.**

1. Комутативність кон'юнкції та диз'юнкції

$$x \cdot y = y \cdot x; x \vee y = y \vee x.$$

2. Асоціативність кон'юнкції та диз'юнкції

$$x \cdot (y \cdot z) = (x \cdot y) \cdot z; x \vee (y \vee z) = (x \vee y) \vee z.$$

3. Дистрибутивність кон'юнкції та диз'юнкції відносно одна одної
 $x \cdot (y \vee z) = (x \cdot y) \vee (x \cdot z); x \vee (y \cdot z) = (x \vee y) \cdot (x \vee z).$

4. Ідемпотентність кон'юнкції та диз'юнкції

$$x \cdot x = x; x \vee x = x.$$

5. Закон виключеного третього

$$x \vee \bar{x} = 1.$$

6. Закон протиріччя

$$x \cdot \bar{x} = 0.$$

7. Закон тотожності з константами

$$x \cdot 0 = 0; x \cdot 1 = x; x \vee 0 = x; x \vee 1 = 1.$$

8. Закон елімінації

$$x \cdot (x \vee y) = x; x \vee (x \cdot y) = x.$$

9. Закон подвійного заперечення

$$\bar{\bar{x}} = x.$$

10. Закон де Моргана

$$\overline{x \cdot y} = \bar{x} \vee \bar{y}; \overline{x \vee y} = \bar{x} \cdot \bar{y}.$$

Способи задання логічних функцій та їх використання для синтезу комп'ютерних схем

Для опису функцій алгебри логіки використовують різні способи. Основними з них є опис функцій у словесній формі, у вигляді таблиць істинності, алгебраїчних виразів та послідовностей десяткових чисел.

Словесний опис функцій алгебри логіки найчастіше застосовують для початкового опису поведінки комп'ютерного пристрою.

Опис функцій алгебри логіки у вигляді таблиці істинності

Означення 1.3.5. Таблицею істинності називають таблицю, в якій кожному інтерпретації логічної функції поставлено у відповідність її значення, рис. 1.3.1.

x_1	x_2	...	x_n	$f(x_1, x_2, \dots, x_n)$
0	0	0	0	0
1	0	0	0	
0	1	0	0	
...	...	...	...	
1	1	1	1	1

Рис. 1.3.1

В таблиці істинності, рис. 1.3.1, кожній змінній та значенню самої функції відповідає по одному стовпчику, а кожній інтерпретації — по одному рядку.

Кількість рядків таблиці відповідає кількості різних інтерпретацій логічної функції. Загалом таблиця істинності містить 2^n рядків.

Приклад 1.3.1. Укласти таблицю істинності для функцій, які дорівнюють «1», якщо два її аргументи теж дорівнюють «1».

Розв'язання. У відповідності з означенням 1.3.7. і рис. 1.3.1, таблиця істинності матиме вигляд, наведений у табл. 1.3.2.

Таблиця 1.3.2

x_2	x_1	x_0	$f(x)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1

x_2	x_1	x_0	$f(x)$
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Опис функцій алгебри логіки у вигляді алгебраїчного виразу

Елементарна кон'юнкція утворюється кон'юнкцією скінченної множини логічних змінних і їх заперечень, наприклад: $f(x, y, z) = x \cdot y \cdot \bar{z}$.

Елементарна диз'юнкція утворюється диз'юнкцією скінченної множини логічних змінних і їх заперечень, наприклад: $f(x, y, z) = x \vee \bar{y} \vee z$.

Кількість змінних в елементарній кон'юнкції (диз'юнкції) називають її довжиною, яка визначає її ранг. Наприклад, $f(x, y, z, t) = \bar{x} \vee y \vee \bar{z} \vee t$ є диз'юнкцією четвертого рангу.

Диз'юнкцією будь-якого числа елементарних кон'юнкцій називають **диз'юнктивною нормальною формою (ДНФ)**, наприклад:

$$\bar{x} \vee y \cdot \bar{z} \vee x \cdot \bar{y} \cdot \bar{z} \vee \bar{x} \cdot y \cdot \bar{z}.$$

Кон'юнкцією будь-якого числа елементарних диз'юнкцій називають **кон'юнктивною нормальною формою (КНФ)**, наприклад:

$$x \cdot (x \vee y) \cdot (y \vee \bar{z}) \cdot (x \vee \bar{y} \vee z).$$

Логічну функцію, задану будь-яким аналітичним виразом, можна безпосередньо перетворити на **нормальну диз'юнктивну або кон'юнктивну форму**. Для цього потрібно:

1. Виразити всі операції через операції кон'юнкції, диз'юнкції та заперечення;

2. Позбутися заперечення над цілими виразами, перейшовши до форми, в якій є заперечення тільки окремих змінних.

3. Розкрити дужки, застосувавши закон дистрибутивності.

4. Звести кон'юнкції (диз'юнкції) до елементарних операцій.

Якщо до складу логічної функції входять набори елементарних кон'юнкцій однакового рангу, пов'язані диз'юнкцією, то таку форму подання логічної функції називають **досконалою диз'юнктивною нормальною формою (ДДНФ)**. **Правила утворення ДДНФ** — функції для n аргументів є такі:

1. За кожним набором двійкових змінних, за яких функція набуває значення 1, скласти елементарні кон'юнкції.

2. В елементарну кон'юнкцію записати неінвертованими змінні, що задані одиницею в таблиці істинності, а інвертованими — ті змінні, які в таблиці істинності задані нулем. Здобутий результат називають конститuentами одиниці.

3. Елементарні кон'юнкції об'єднати знаком диз'юнкції.

Досконалою кон'юнктивною нормальною формою (ДКНФ) логічної функції називають такий її вираз, який містить елементарні диз'юнкції одного рангу, пов'язані кон'юнкцією. **Правила утворення ДКНФ** для n аргументів є такі:

1. За кожним набором двійкових змінних, за яких функція набуває значення 0, скласти елементарні диз'юнкції (конституенти нуля).

2. В елементарні диз'юнкції записати неінвертованими змінні, що задані нулем в таблиці істинності, а інвертованими — ті змінні, які в таблиці істинності задані одиницею. Здобуті суми називають конститuentами нуля.

3. Елементарні диз'юнкції об'єднати знаком кон'юнкції.

Приклад 1.3.2. Таблицею істинності, табл. 1.3.2, задана функція $f(x, y, z)$. Користуючись заданою таблицею, потрібно утворити її ДДНФ і ДКНФ.

Розв'язання. Використовуючи означення 1.3.5, а також правила утворення ДДНФ і ДКНФ, отримаємо відповідні конституенти «1» і конституенти «0», що наведені в таблиці 1.3.3.

Таблиця 1.3.3

Значення аргументу			Значення функції $f(x, y, z)$	ДДНФ	ДКНФ
x	y	z		Конституенти «1»	Конституенти «0»
0	0	0	0	—	$x \vee y \vee z$
0	0	1	1	$\bar{x} \cdot \bar{y} \cdot z$	—
0	1	0	1	$\bar{x} \cdot y \cdot \bar{z}$	—
0	1	1	0	—	$x \vee \bar{y} \vee \bar{z}$
1	0	0	1	$x \cdot \bar{y} \cdot \bar{z}$	—
1	0	1	0	—	$\bar{x} \vee y \vee \bar{z}$
1	1	0	0	—	$\bar{x} \vee \bar{y} \vee z$
1	1	1	1	$x \cdot y \cdot z$	—

Функція конституент «1» матиме вигляд:

$f(x, y, z) = \bar{x} \cdot \bar{y} \cdot z \vee \bar{x} \cdot y \cdot \bar{z} \vee x \cdot \bar{y} \cdot \bar{z} \vee x \cdot y \cdot z$, а функція конституент «0»

$$f(x, y, z) = (x \vee y \vee z) \cdot (x \vee \bar{y} \vee \bar{z}) \cdot (\bar{x} \vee y \vee \bar{z}) \cdot (\bar{x} \vee \bar{y} \vee z).$$

Від будь-якої ДНФ можна перейти до ДДНФ функції за допомогою рівнозначних перетворень. Такий перехід називають **розгортанням**. Для цього потрібно:

1. Ввести відсутні змінні в кожен кон'юнкцію множенням її на рівнозначність виразу $x \vee \bar{x} = 1$, де x — відсутня змінна.

2. Розкрити дужки, застосувавши комутативний закон $x \cdot y = y \cdot x$.
3. Позбутися кон'юнкцій, що повторюються, використовуючи закон ідемпотентності $x \cdot y \vee x \cdot y = x \cdot y$.

Приклад 1.3.3. Для заданої ДНФ функції $f(x, y, z) = x \cdot \bar{y} \cdot z \vee x \cdot \bar{z}$, знайти її ДДНФ.

Розв'язання. Використовуючи правило переходу до ДДНФ, отримаємо,

$$f(x, y, z) = x \cdot \bar{y} \cdot z \vee x \cdot \bar{z} \cdot (y \vee \bar{y}) = x \cdot \bar{y} \cdot z \vee x \cdot y \cdot \bar{z} \vee x \cdot \bar{y} \cdot \bar{z}.$$

Перехід від КНФ до ДКНФ здійснюють аналогічно переходу від ДНФ до ДДНФ. Для цього необхідно:

1. Ввести відсутні змінні в кожну диз'юнкцію, використавши закон протиріччя $x \cdot \bar{x} = 0$, де x — відсутня змінна.
2. Здійснити перетворення, застосувавши закон дистрибутивності $x \vee y \cdot z = (x \vee y) \cdot (x \vee z)$ і комутативний закон $x \vee y = y \vee x$.
3. Позбутися диз'юнкцій, що повторюються, на основі закону ідемпотентності.

Приклад 1.3.4. Для заданої КНФ функції $f(x, y, z) = (x \vee y) \cdot (\bar{y} \vee z) \cdot (\bar{x} \vee \bar{z})$ знайти її ДКНФ.

Розв'язання. Використовуючи правило переходу до ДКНФ, отримаємо:

$$\begin{aligned} f(x, y, z) &= (x \vee y \vee z \cdot \bar{z}) \cdot (\bar{y} \vee z \vee x \cdot \bar{x}) \cdot (\bar{x} \vee \bar{z} \vee y \cdot \bar{y}) = \\ &= (x \vee y \vee z) \cdot (x \vee y \vee \bar{z}) \cdot (x \vee \bar{y} \vee z) \cdot (\bar{x} \vee \bar{y} \vee z) \cdot \\ &\quad \cdot (\bar{x} \vee y \vee \bar{z}) \cdot (\bar{x} \vee y \vee \bar{z}) \end{aligned}$$

Опис функцій алгебри логіки у вигляді послідовності десяткових чисел

Іноді для скорочення запису функцію алгебри логіки її зображують у вигляді послідовності десяткових чисел, при цьому послідовно записують десяткові еквіваленти двійкових кодів відповідних конститuent «1» або «0».

Приклад 1.3.5. Записати у вигляді послідовності чисел функцію $f(x, y, z)$ з прикладу 1.3.3 та прикладу 1.3.4.

Розв'язання. У ДДНФ перша конститuent «1» відповідає двійковому коду 101, друга — двійковому коду 110, а третя — двійковому коду 100. Десятковий еквівалент цих кодів відповідно дорівнює: 5, 6, 4. Аналогічно для ДКНФ конституенти «0» матимуть наступні десяткові еквіваленти: 0, 1, 2, 5, 6, 7. Виходячи із цього, дані функції можна записати в наступному вигляді для ДДНФ $f(x, y, z) = \vee(4, 5, 6)$ і для ДКНФ $f(x, y, z) = \wedge(0, 1, 2, 5, 6, 7)$.

Мінімізація логічних функцій

Завдання мінімізації логічних функцій полягає у пошуку найпростішої формули логічної функції за обраним критерієм. В якості критерію може виступати кількість змінних у формулі, кількість знаків кон'юнкції та диз'юнкції або комбінація подібних критеріїв. Мінімальні форми, які отримані в процесі мінімізації називають мінімальними ДНФ і КНФ.

Означення 1.3.6. Імплікантою деякої логічної функції f називають таку функцію ϕ , що на всіх інтерпретаціях, на яких вона дорівнює одиниці, ϕ теж дорівнює одиниці. Елементарні кон'юнкції, що входять до складу ДНФ функції, також є її імплікантами.

Означення 1.3.7. Простою імплікантою логічної функції f називають таку імпліканту, що ніяка її власна частина не є імплікантою даної функції.

Означення 1.3.8. Скороченою ДНФ називають диз'юнкцію всіх простих імплікант логічної функції.

Означення 1.3.9. Тупиковою ДНФ називають ДНФ даної логічної функції, яка складається тільки з простих імплікант.

На відміну від скороченої ДНФ, тупикова ДНФ може не містити деякі з простих імплікант функції. Кожна логічна функція має єдину скорочену ДНФ і може мати декілька тупикових ДНФ.

Означення 1.3.10. Мініальною ДНФ (МДНФ) логічної функції називають одну з тупикових ДНФ, якій відповідає найменше значення критерію мінімізації ДНФ.

Якщо логічна функція задана ДНФ, то для знаходження її простих імплікант використовують наступні операції для перетворення формул алгебри логіки.

Операція неповного диз'юнктивного склеювання:

$$A \cdot x \vee A \cdot \bar{x} = A \vee A \cdot x \vee A \cdot \bar{x}.$$

Операція диз'юнктивного поглинання: $A \vee A \cdot x = A$.

Операція повного диз'юнктивного склеювання: $A \cdot x \vee A \cdot \bar{x} = A$, де A — деяка елементарна кон'юнкція змінних, а x — логічна змінна.

Приклад 1.3.6. Виконати операції повного склеювання логічної функції заданої в ДДНФ різними способами:

$$f(x, y, z) = x \cdot y \cdot z \vee \bar{x} \cdot y \cdot z \vee x \cdot \bar{y} \cdot z \vee \bar{x} \cdot \bar{y} \cdot z.$$

Розв'язання. Виконаємо операції повного склеювання першим способом:

$$\begin{aligned} f(x, y, z) &= x \cdot y \cdot z \vee \bar{x} \cdot y \cdot z \vee x \cdot \bar{y} \cdot z \vee \bar{x} \cdot \bar{y} \cdot z = (x \cdot y \cdot z \vee \bar{x} \cdot y \cdot z) \vee \\ &\vee (x \cdot \bar{y} \cdot z \vee \bar{x} \cdot \bar{y} \cdot z) \vee (x \cdot y \cdot z \vee \bar{x} \cdot y \cdot z) \vee (x \cdot \bar{y} \cdot z \vee \bar{x} \cdot \bar{y} \cdot z) = y \cdot z \vee \bar{x} \cdot z \vee x \cdot \bar{y}. \end{aligned}$$

Тепер виконаємо операції склеювання другим способом, інакше компонуючи імпліканти:

$$f(x, y, z) = (x \cdot y \cdot z \vee \bar{x} \cdot y \cdot z) \vee (x \cdot \bar{y} \cdot z \vee \bar{x} \cdot \bar{y} \cdot z) = y \cdot z \vee \bar{x} \cdot z.$$

В результаті склеювання отримані дві різні тупикові ДНФ логічної функції $f(x, y, z)$. Друга тупикова ДНФ простіша за першу, оскільки містить меншу кількість символів змінних і знаків операцій.

Означення 1.3.11. Імпліцентию деякої логічної функції f називають таку функцію ϕ , яка на всіх інтерпретаціях, де вона дорівнює нулю, функція f теж дорівнює нулю. Елементарні диз'юнкції, що входять до складу КНФ функції, також є її імпліцентами.

Означення 1.3.12. Простою імпліцентию логічної функції f називають таку імпліценту, що ніяка її власна частина не є імпліцентию даної функції.

Означення 1.3.13. Скороченою КНФ називають кон'юнкцію всіх простих імпліцент логічної функції.

Означення 1.3.14. Тупиковою КНФ називають КНФ даної логічної функції, що складається тільки з простих імпліцент.

Означення 1.3.15. Мінімальною КНФ (МКНФ) логічної функції називають одну із тупикових КНФ, якій відповідає найменше значення критерію мінімізації КНФ.

Для мінімізації КНФ використовують такі операції склеювання.

Операція неповного кон'юнктивного склеювання:

$$(A \vee x) \cdot (A \vee \bar{x}) = A \cdot (A \vee x) \cdot (A \vee \bar{x}).$$

Операція кон'юнктивного поглинання $A \cdot (A \vee x) = A$.

Операція повного кон'юнктивного поглинання $(A \vee x) \cdot (A \vee \bar{x}) = A$, де A — деяка елементарна диз'юнкція, а x — логічна змінна.

При мінімізації логічних функцій використовують методи Карно і Квайна, що набули найбільшого поширення. Метод Карно має структуру таблиць, яка приведена на рис. 1.3.2.

a)

	x	0	1
y	0		
	1		

б)

	xy	00	01	11	10
z	0				
	1				

в)

	xy	00	01	11	10
zt	00				
	01				
	11				
	10				

Рис. 1.3.2

На рис 1.3.2а наведена таблиця Карно для двох змінних, на рис. 1.3.2б. — для трьох, а на рис. 1.3.2в — для чотирьох змінних. Значення змінних у таблицях Карно розташовані у заголовках рядків і стовпчиків. Кожній конституенті одиниці або нуля функції відповідає одна клітинка таблиці. Нуль або одиниця в клітинці визначає значення функції на даній інтерпретації. Значення змінних розта-

шовані так, щоб сусідні рядки і стовпчики таблиці відрізнялися значенням тільки однієї змінної. Склеювання клітинок у таблиці Карно і отримання мінімальної ДНФ відбувається за наступним правилом.

1. Клітинки об'єднуються у групи, що позначають операції склеювання. В об'єднанні беруть участь тільки ті сусідні клітинки, в яких розміщені одиниці.

2. В групу дозволяється об'єднувати кількість клітинок 2^n , $n = 1, 2, 3 \dots$. При цьому група може мати лише прямокутну або квадратну форму.

3. При склеюванні необхідно знайти набір максимальних груп клітинок. Під максимальною групою розуміють групу, яка не входить цілком у жодну іншу групу і відповідає простій імпліканті функції. Кількість груп у такому наборі повинна бути мінімальною, оскільки така група відповідає мінімальній тупиковій ДНФ. Кожна одиниця таблиці Карно повинна входити хоча б до однієї групи, що забезпечує покриття функції отриманим набором імплікант.

4. Кожна група клітинок, що отримана після склеювання, відповідає тій імпліканті функції, реальні змінні якої мають однакове значення для всіх клітинок групи.

5. Диз'юнкція всіх отриманих простих імплікант зображує результат мінімізації формули і є мінімальною ДНФ.

Оскільки при мінімізації на множині ДНФ у склеюванні беруть участь тільки клітинки, що містять одиниці, то нулі в таблиці Карно, як правило, не вказують, а мають на увазі, що порожні клітинки містять нулі.

Приклад 1.3.7. Знайти МДНФ для функції

$$f(x, y, z) = x \cdot y \cdot \bar{z} \vee x \cdot \bar{y} \cdot \bar{z} \vee x \cdot \bar{y} \cdot z \vee \bar{x} \cdot y \cdot \bar{z} \vee \bar{x} \cdot y \cdot z$$

Розв'язання. У відповідності з рис. 1.3.2б, будуємо таблицю Карно, до якої заносимо одиниці згідно із заданою логічною функцією, рис. 1.3.3.

xy z	00	01	11	10
0	1	1	1	1
1				1

Рис. 1.3.3

Виконуємо склеювання клітинок таблиці у відповідності з розглянутими вище правилами. При склеюванні об'єднаємо чотири клітинки, де містяться чотири одиниці першої стрічки, які накладаються змінною $\bar{z}$, а одиниця, яка залишилась в останньому стовпчику, склеюється з одиницею останнього стовпчика першої стрічки. У результаті цього отримаємо МДНФ логічної функції $f(x, y, z) = \bar{z} \vee x \cdot y$. Знаходження МКНФ виконують аналогічно описаному вище, але тільки замість одиниць у клітинках таблиці записують нулі.

Метод Квайна

Метод мінімізації Квайна реалізує перехід від ДДНФ (ДКНФ) до ДНФ (КНФ) із використанням операцій склеювання та поглинання.

Основу алгоритма Квайна складають такі кроки.

1. Записати ДДНФ (ДКНФ) заданої функції.

2. Виконати всі можливі операції неповного диз'юнктивного (кон'юнктивного) склеювання. Отримана формула є диз'юнкцією (кон'юнкцією) всіх можливих імплікант (імпліцент) даної функції.

3. Виконати всі можливі операції диз'юнктивного (кон'юнктивного) поглинання. Підсумкова формула є скороченою ДНФ (КНФ) даної функції.

4. Скласти імплікантну (імпліцентну) таблицю і шляхом її покриття за рахунок мінімальної кількості простих імплікант (імпліцент) знайти всі тупикові ДНФ (КНФ) даної логічної функції.

5. Серед тупикових ДНФ (КНФ) знайти мінімальну ДНФ (КНФ) логічної функції.

Приклад 1.3.8. Знайти методом Квайна МДНФ функції

$$f(x, y, z) = x \cdot y \cdot z \vee x \cdot y \cdot \bar{z} \vee \bar{x} \cdot y \cdot z \vee \bar{x} \cdot y \cdot \bar{z} \vee x \cdot \bar{y} \cdot \bar{z} \vee \bar{x} \cdot \bar{y} \cdot \bar{z}$$

Розв'язання. Виконаємо всі можливі операції диз'юнктивного склеювання і поглинання

$$x \cdot y \cdot z \vee x \cdot y \cdot \bar{z} = x \cdot y; \quad x \cdot y \cdot z \vee \bar{x} \cdot y \cdot z = y \cdot z;$$

$$x \cdot y \cdot \bar{z} \vee \bar{x} \cdot y \cdot \bar{z} = \bar{x} \cdot \bar{z}; \quad \bar{x} \cdot y \cdot z \vee \bar{x} \cdot y \cdot \bar{z} = \bar{x} \cdot y;$$

$$x \cdot y \cdot z \vee x \cdot \bar{y} \cdot \bar{z} = x \cdot \bar{z}; \quad x \cdot y \cdot z \vee \bar{x} \cdot \bar{y} \cdot \bar{z} = \bar{y} \cdot \bar{z}.$$

В результаті цього отримаємо формулу

$$f(x, y, z) = x \cdot y \cdot z \vee x \cdot y \cdot \bar{z} \vee \bar{x} \cdot y \cdot x \vee \bar{x} \cdot y \cdot z \vee x \cdot \bar{y} \cdot \bar{z} \vee \bar{x} \cdot \bar{y} \cdot \bar{z} = \\ = x \cdot y \vee y \cdot z \vee x \cdot z \vee \bar{x} \cdot z \vee \bar{x} \cdot y \vee \bar{y} \cdot z.$$

Із аналізу формули випливає, що отримані інші елементарні кон'юнкції не підлягають подальшому склеюванню і тому диз'юнкція всіх можливих імплікант даної функції матиме вигляду

$$f(x, y, z) = x \cdot y \vee y \cdot z \vee x \cdot z \vee \bar{x} \cdot z \vee \bar{x} \cdot y \vee \bar{y} \cdot z.$$

Отримана формула є скороченою ДНФ заданої функції. Для знаходження МДНФ складемо імплікантну таблицю, табл. 1.3.4.

Таблиця 1.3.4

	$x \cdot y \cdot z$	$x \cdot y \cdot \bar{z}$	$\bar{x} \cdot y \cdot z$	$\bar{x} \cdot y \cdot \bar{z}$	$x \cdot \bar{y} \cdot z$	$\bar{x} \cdot \bar{y} \cdot z$
$x \cdot y$	*	*				
$y \cdot z$	*		*			
$x \cdot \bar{z}$		*			*	
$\bar{x} \cdot y$			*	*		
$\bar{x} \cdot \bar{y}$				*		*
$\bar{y} \cdot z$					*	*

Рядки заданої таблиці є простими імплікантами, а стовпчики-конституенти одиницями досліджуваної функції. Зірочкою в таблиці відмічається кожна клітинка, для якої імпліканта рядка є власною частиною конституенти зі стовпчика. Користуючись кроком чотири алгоритму, шляхом покриття за рахунок мінімальної кількості простих імплікант, знаходимо дві тупикові ДНФ, що мають наступний вигляд

$$\text{ДНФ1: } f(x, y, z) = x \cdot y \vee \bar{x} \cdot z \vee \bar{y} \cdot \bar{z};$$

$$\text{ДНФ2: } f(x, y, z) = y \cdot z \vee x \cdot \bar{z} \vee \bar{x} \cdot y.$$

Вказані ДНФ містять по 6 символів змінних, а також однакову кількість знаків операцій диз'юнкції (2), кон'юнкції (3) і заперечення (3). В якості МДНФ можна вибрати ДНФ1 або ДНФ2. Знаходження МКНФ виконують аналогічно описаному вище.

1.4. Автоматні основи комп'ютерної схемотехніки

Основні визначення

Теорія автоматів має справу з математичними моделями, які використовують, наприклад, для розробки вузлів, схем та пристроїв комп'ютерної схемотехніки.

Означення 1.4.1. Автоматом називають певну систему, яка має n — входів і m — виходів і яка робить відповідне перетворення множини вхідних сигналів $A = \{a_1, a_2, \dots, a_n\}$ в множину вихідних сигналів $B = \{b_1, b_2, \dots, b_m\}$, рис. 1.4.1а.

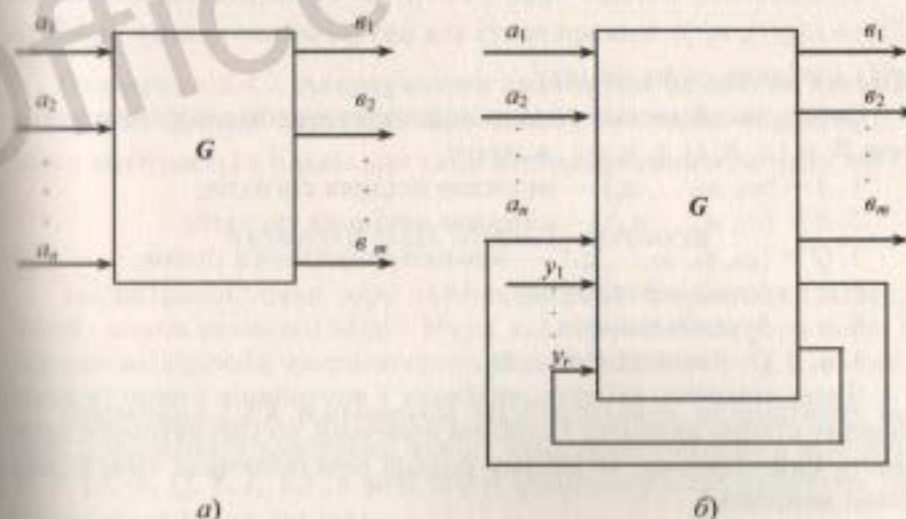


Рис. 1.4.1

Якщо характер перетворень всередині автомата в дискретні моменти часу $t = 0, 1, 2, \dots$ залежить тільки від комбінації сигналів, які поступають на його входи, то такі автомати називають комбінаційними. У таких автоматах вихідні сигнали є функцією вхідних сигналів, тобто $B_t = G(a_t)$.

Якщо характер перетворень всередині автомата в дискретні моменти часу $t = 0, 1, 2, \dots$ залежить не тільки від вхідних сигналів, але і від сигналів, що прийшли до автомата раніше, то такий автомат називають **автоматом з пам'яттю** (в подальшому просто автомат), рис. 1.4.1б.

В таких автоматах вихідні сигнали залежать не тільки від множини вхідних сигналів $A = \{a_1, a_2, \dots, a_n\}$, але і від множини внутрішніх станів автомата $Q = \{q_0, q_1, q_2, \dots, q_s\}$. В них вихідні сигнали та внутрішні стани є функціями станів автомата і множини вхідних сигналів. Відображення, що індукується таким автоматом, однозначно визначається заданням у ньому двох функцій, які називають:

1. Функцією переходів $\delta(q(t), a(t))$, яка визначає внутрішні стани автомата згідно із значення вхідних сигналів $a(t)$ і попереднього стану автомата $q(t-1)$;

2. Функцією виходів $\lambda(q(t), a(t))$, що визначає вихідні сигнали $Z(t) = \lambda(q(t), a(t))$ в залежності від внутрішнього стану автомата $q(t)$ і вхідних сигналів $a(t)$.

Тобто, такий автомат можна задати шестикомпонентним вектором $W = \{A, B, Q, \delta, \lambda, q_0\}$, в якого:

1. $A = \{a_1, a_2, \dots, a_n\}$ — множина вхідних сигналів;
2. $B = \{b_1, b_2, \dots, b_m\}$ — множина вихідних сигналів;
3. $Q = \{q_0, q_1, q_2, \dots, q_s\}$ — множина внутрішніх станів;
4. δ — функція переходів;
5. λ — функція виходів;
6. $q_0 \in Q$ — початковий стан.

Якщо кількість вхідних, вихідних і внутрішніх станів (у подальшому станів) автомата є кінцева величина, то такі автомати називають **скінченими**. В даному розділі розглядаються тільки скінченні автомати.

Означення 1.4.2. Абстрактним автоматом називають математичну модель дискретного пристрою задану шестикомпонентним вектором $W = \{A, B, Q, \delta, \lambda, q_0\}$, яка абстрагується від його змісту в частині визначення природи внутрішньої структури, вхідних і вихідних сигналів.

Означення 1.4.3. Структурним автоматом називають математичну модель дискретного пристрою задану шестикомпонентним

вектором $W = \{A, B, Q, \delta, \lambda, q_0\}$, в якій визначені природа і структура побудови внутрішніх станів, вхідних та вихідних сигналів.

Означення 1.4.4. Повністю визначеним автоматом називають автомат, у якого ділянки визначення переходів $\delta(q(t), a(t))$ і виходів $\lambda(q(t), a(t))$ співпадають з множиною $A \times B$ усіх пар виду (a_i, b_j) , де $i \in n, j \in m$:

Означення 1.4.5. Частково визначеним автоматом називають автомат, у якого ділянки визначення функції переходів $\delta(q(t), a(t))$ і виходів $\lambda(q(t), a(t))$ визначені не для всіх пар $(a_i, b_j) \in A \times B$, де $i \in n, a j \in m$.

Означення 1.4.6. Синхронним автоматом називають автомат, робота якого, тобто виконання функції переходів $\delta(q(t), a(t))$ і виходів $\lambda(q(t), a(t))$, відбувається через рівні інтервали часу $t = 0, 1, 2, \dots$.

Означення 1.4.7. Асинхронним автоматом називають автомат, виконання функції переходів $\delta(q(t), a(t))$ і виходів $\lambda(q(t), a(t))$ в якого відбувається тільки при зміні вектора вхідного сигналу $a(t)$.

Автомати Мілі, Мура, С-автомати

З практичної точки зору найбільше розповсюдження (застосування) мають автомати Мілі і Мура, які отримали назву на честь їх перших дослідників, американських учених G.H. Mealy і E.F. Moore.

Означення 1.4.8. Автоматом Мілі називають математичну модель дискретного пристрою, задану шестикомпонентним вектором $W = \{A, B, Q, \delta, \lambda, q_0\}$, в якій закон функціонування описують наступною системою рівнянь

$$\begin{cases} q(t+1) = \delta(q(t), a(t)); \\ 2(t) = \lambda(q(t), a(t)), \end{cases} \quad (1.4.1)$$

де $\delta(q(t), a(t))$ — функція переходів;
 $\lambda(q(t), a(t))$ — функція виходів;
 $q(t+1)$ — стан системи, в який вона переходить зі стану $q(t)$ під дією вхідних сигналів $a(t)$;

$v(t)$ — вихідний сигнал, який система видає при переходу її із стану $q(t)$ в стан $q(t+1)$ під дією вхідних сигналів $a(t)$.

Означення 1.4.9. Автоматом Мура називають математичну модель дискретного пристрою задану шестикомпонентним вектором $W = \{A, B, Q, \delta, \lambda, q_0\}$, в якій закон функціонування описують наступною системою рівнянь:

$$\left. \begin{aligned} q(t+1) &= \delta(q(t), a(t)); \\ Z(t) &= \lambda(q(t)), \end{aligned} \right\} \quad (1.4.2)$$

де $Z(t)$ — вихідний сигнал, який система видає при знаходженні її в стані $q(t)$.

Означення 1.4.10. С-автоматом називають математичну модель дискретного пристрою задану восьмикомпонентним вектором $W = \{A, B, Z, Q, \lambda_1, \lambda_2, \delta, q_0\}$, в якій закон функціонування описують наступною системою рівнянь

$$\left. \begin{aligned} q(t+1) &= \delta(q(t), a(t)); \\ b(t) &= \lambda_1(q(t), a(t)), \\ Z(t) &= \lambda_2(q(t)) \end{aligned} \right\} \quad (1.4.3)$$

де $B = \{v_1, v_2, \dots, v_m\}$ — множина вихідних сигналів типу 1;

$Z = \{z_1, z_2, \dots, z_p\}$ — множина вихідних сигналів типу 2;

$\lambda_1(q(t), a(t))$ — функція виходів, яку реалізує автомат Мілі;

$\lambda_2(q(t))$ — функція виходів, що реалізує автомат Мура.

Із закону функціонування автомата Мілі (система рівнянь 1.4.1) випливає, що його вихідний сигнал $v(t) = \lambda_1(q(t), a(t))$ видається в час дії вхідного сигналу $a(t)$ і перебуванні автомата в стані $q(t)$, а автомата Мура (система рівняння 1.4.2), вихідний сигнал $Z(t) = \lambda_2(q(t))$ видається весь час, поки автомат перебуває в стані $q(t)$. В подальшому для кращого розуміння і перетворення автоматів буде використовуватись також сигнал $v(t)$ в якості вихідного сигналу $Z(t)$.

Способи задання автоматів

Для задання автоматів застосовують три способи: табличний, графічний і за допомогою матриці переходів. Серед них найбільше практичне використання знайшли табличний і графічний способи.

Опис роботи повністю визначеного автомата Мілі табличним способом за допомогою таблиць переходів і виходів ілюструється на прикладі автомата W_1 наведено в табл. 1.4.1 і табл. 1.4.2.

Таблиця 1.4.1

$q_i \backslash q_j$	q_0	q_1	q_2
a_1	q_2	q_2	q_1
a_2	q_1	q_1	q_0

Таблиця 1.4.2

$q_i \backslash q_j$	q_0	q_1	q_2
a_1	v_1	v_1	v_3
a_2	v_2	v_3	v_2

Стрічки цих таблиць відповідають вхідним сигналам, а стовпчики — станам. Крайній зліва стовпчик позначений початковим станом q_0 . На пересіченні стовпчика q_j і стрічки a_i в таблиці переходів ставиться стан $q_s = \delta(a_i, q_j)$, в який автомат переходить із стану q_j у стан q_s , а в таблиці виходів — відповідний цьому переходу вихідний сигнал $v_s = \lambda(a_i, q_j)$. Оскільки в автоматі Мура вихідний сигнал залежить тільки від стану, то цей автомат задається однією відміченою таблицею переходів, в якій кожному її стовпчику приписаний, крім стану q_j , ще і вихідний сигнал $v_s = \lambda(q_j)$, відповідний даному стану. Приклад табличного задання повністю визначеного автомата Мура W_2 , приведений у відміченій таблиці переходів, табл. 1.4.3.

Таблиця 1.4.3

$q_i \backslash v_k$	v_0	v_1	v_2	v_3	v_4
q_j	q_0	q_1	q_2	q_3	q_4
a_i	q_1	q_0	q_3	q_1	q_3
a_2	q_2	q_3	q_4	q_0	q_2

Для часткового визначеного автомата Мілі, в якого функції переходів $\delta(a_i(t), q_j(t))$ і виходів $\lambda(a_i(t), q_j(t))$ визначені не для всіх пар $(a_i, v_j) A \times B$, де $i \in n$, а $j \in m$, то на місцях невизначених станів і вихідних сигналів ставлять прочерк. Аналогічно табличним способом задають і частково визначені автомати Мура.

Графічний спосіб задання автоматів Мілі полягає в наступному. Стани автоматів зображують вершинами графа, що з'єднують дугами. Дві вершини графа автомата q_0 і q_s (початковий стан і стан переходу) з'єднують дугою, направленою від q_0 до q_s , якщо в авто-

маті є перехід від q_0 до q_k , тобто якщо $q_k = \delta(q_0(t), a_i(t))$, при деякому $a_i \in A$. Дузі (q_0, q_k) графа автомата приписують вихідний сигнал $e_j = \lambda(q_0(t), a_i(t))$, якщо він визначений, і роблять прочерк — у протилежному випадку. Якщо перехід автомата зі стану q_0 до стану q_k відбувається під дією декількох вхідних сигналів, то тоді дузі (q_0, q_k) для автомата Мілі приписують усі ці вхідні і відповідні вихідні сигнали, рис. 1.4.2а.

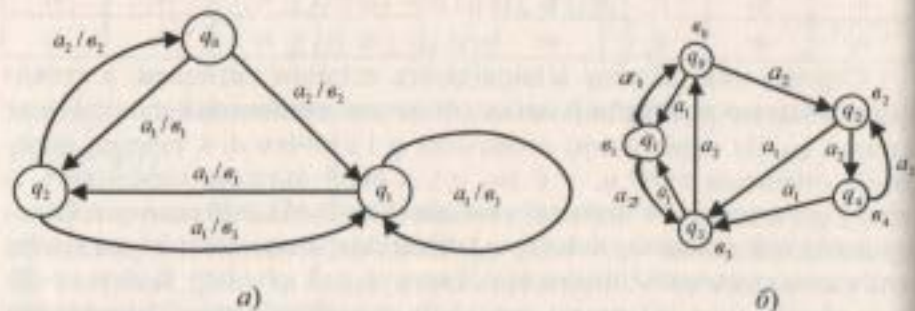


Рис. 1.4.2

При заданні автомата Мура за допомогою графа вихідний сигнал $e_k = \lambda(q_k)$ записують, як правило, поруч з його станом, рис. 1.4.2б.

Канонічний метод структурного синтезу автоматів

Канонічний метод структурного синтезу розглянемо для C — автомата, оскільки ця модель є об'єднанням моделі Мілі та Мура, рис. 1.4.3.

Структурна схема C — автомата складається з трьох частин: пам'яті і двох комбінаційних схем KC_1 та KC_2 . Якщо необхідно синтезувати автомат Мілі, то в C — автоматі функцію λ_2 не задають і буде відсутня комбінаційна схема KC_2 . У випадку моделі Мура не заданою буде функція λ_1 і в комбінаційній схемі KC_1 будуть відсутні вихідні сигнали $e_1 \dots e_m$.

Комбінаційна схема KC_1 слугує формуванню вихідних сигналів типу 1 і вхідних сигналів для автоматів пам'яті, а KC_2 — формуванню сигналів типу 2.

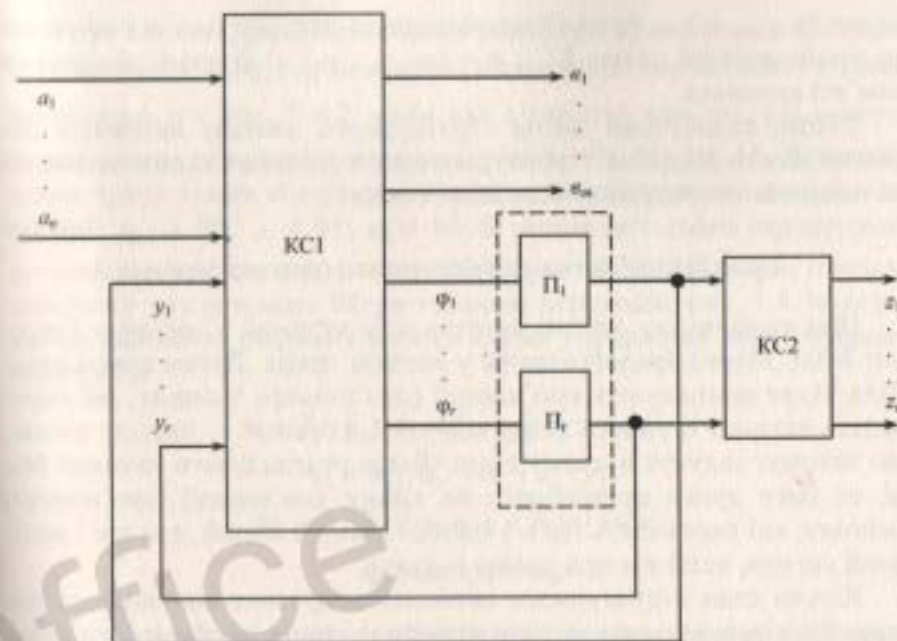


Рис. 1.4.3

Пам'ять автомата складається з вибраних автоматів пам'яті — елементарних автоматів Мура $P_1, \dots, P_n$. Після вибору елементів пам'яті кожний стан абстрактного C — автомата кодується в структурному автоматі. Якщо всі автомати $P_1, \dots, P_n$ однакові, то їх число буде дорівнювати

$$n = \lceil \log_{\Theta} M \rceil, \quad (1.4.4)$$

де Θ — число станів елементарного автомата пам'яті;

M — число станів абстрактного C — автомата;

$\lceil \rceil$ — знак того, що береться найбільше ціле число від результату логарифму.

При двійковій системі числення $\Theta = 2$. Таким чином, після вибору елементів пам'яті і кодування станів синтез структурного автомата зводиться до синтезу комбінаційної схеми, яка реалізує функції:

$$\begin{aligned} e_1 &= e_1(a_1, \dots, a_m, y_1, \dots, y_r), \dots, e_m = e_m(a_1, \dots, a_m, y_1, \dots, y_r), \\ \phi_1 &= \phi_1(a_1, \dots, a_m, y_1, \dots, y_r), \dots, \phi_r = \phi_r(a_1, \dots, a_m, y_1, \dots, y_r), \\ z_1 &= z_1(y_1, \dots, y_r), \dots, z_n = z_n(y_1, \dots, y_r), \end{aligned}$$

де $y = (y_1, \dots, y_r)$ — функції зворотного зв'язку від пам'яті автомата до комбінаційної схеми КС₁; $\varphi = (\varphi_1, \dots, \varphi_r)$ — функції збудження пам'яті автомата.

Тобто, канонічний метод структурного синтезу автоматів дозволяє звести завдання структурного синтезу задовільних автоматів до завдання синтезу комбінаційних схем.

Графічний метод структурного синтезу автоматів

При графічному методі синтезу структурний С-автомат (автомат Мілі, Мура) представляють у вигляді графа. Дугам графа автомата Мура приписують кон'юнкції (диз'юнкції) змінних, які переводять автомат із одного стану в другий, а станам — вихідні змінні, що автомат індукує в цьому стані. Якщо розглядувати автомат Мілі, то його дугам приписують не тільки кон'юнкції (диз'юнкції) змінних, які переводять його з одного стану в інший, але ще і вихідний сигнал, який він при цьому індукує.

Кожен стан структурного автомата кодується відповідним кодом. Розмір коду залежить від кількості станів абстрактного автомата і визначається формулою 1.4.4. Для автоматів, що за умовою не є самокорегуючими, як правило, використовують звичайний двійковий код.

Приклад 1.4.1. Розробити схему управління комп'ютерним пристроєм, алгоритм роботи якого заданий абстрактним автоматом,

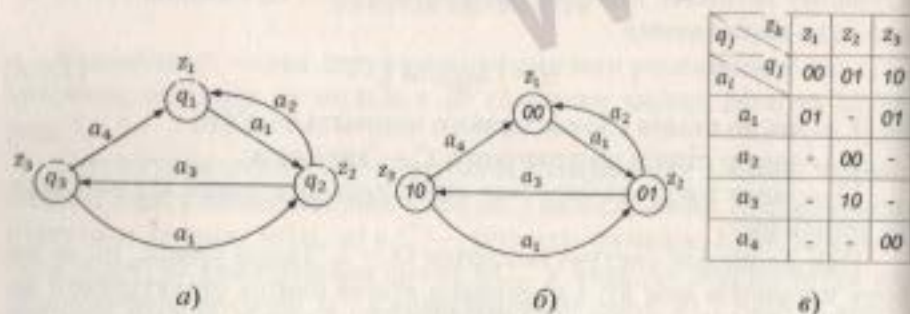


Рис. 1.4.4.

де $a_1, \dots, a_4$ — вхідні змінні пристрою; $z_1, \dots, z_3$ — сигнали, які видає пристрій при управлінні; $q_1, \dots, q_3$ — абстрактні позначення станів, в яких перебуває пристрій при управлінні.

Розв'язання. Для забезпечення реалізації трьох станів абстрактного автомата q_1, q_2, q_3 необхідно в структурному автоматі згідно з формулою $n = \lceil \log_2 3 \rceil = 2$ мати два елементи пам'яті, які можуть задовільнити реалізацію чотирьох станів: 00; 01; 10; 11. Для кодування трьох станів абстрактного автомата використаємо кодові комбінації: $q_1 \rightarrow 00$; $q_2 \rightarrow 01$; $q_3 \rightarrow 10$. В результаті цього структурний автомат матиме вигляд, наведений на рис. 1.4.4б. Використовуючи відмічену для автомата Мура таблицю переходів, рис. 1.4.4в, отримаємо канонічні рівняння роботи схеми управління комп'ютерним пристроєм

$$z_1 = \bar{y}_1 \cdot \bar{y}_2; \quad z_2 = \bar{y}_1 \cdot y_2; \quad z_3 = y_1 \cdot \bar{y}_2;$$

$$\varphi_1^1 = a_3; \quad \varphi_1^0 = a_1 \vee a_4 \cdot \bar{y}_2;$$

$$\varphi_2^1 = a_1 \cdot \bar{y}_1 \vee a_1 = a_1 \cdot (\bar{y}_1 \vee 1) = a_1;$$

$$\varphi_2^0 = a_2 \cdot \bar{y}_1 \vee a_3,$$

де φ_1^1, φ_2^1 і φ_1^0, φ_2^0 — функції включення і виключення відповідно першого і другого елементів пам'яті структурного автомата Мура;

y_1, y_2 і $\bar{y}_1, \bar{y}_2$ — сигнали на виходах першого і другого елементів пам'яті, які відповідають логічним сигналам «1» і «0».

Функція φ_1 відповідає стану коду розряду, розміщеного зліва, а φ_2 — справа. Рівняння включення першого елемента пам'яті φ_1^1 отримують таким чином. У відміченій таблиці переходів розглядають усі переходи кодових станів цієї функції з «0» до «1» під дією вхідних змінних. В кон'юнкцію вхідних змінних також записують і зміну другого елемента пам'яті, якщо вона не міняє свій знак при цьому переході. Якщо цей перехід для функції φ_1^1 відбувається не один раз, а, наприклад, два, то знайдені кон'юнкції змінних об'єднують знаком диз'юнкції.

Рівняння виключення першого елемента пам'яті φ_1^0 отримують аналогічно описаному з тією лише різницею, що при цьому розглядають лише переходи із стану «1» до стану «0». Рівняння для функції φ_2 отримують аналогічно описаному для функції φ_1 .

Схема управління комп'ютерним пристроєм, яка реалізує канонічні рівняння роботи структурного автомата Мура, приведена на рис. 1.4.5.

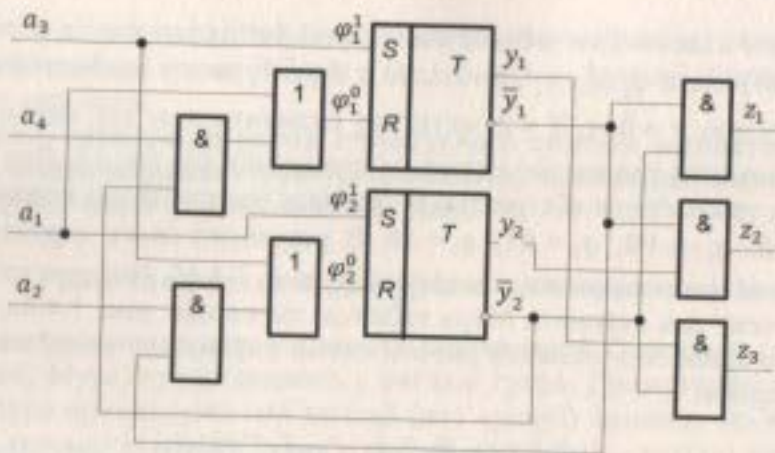
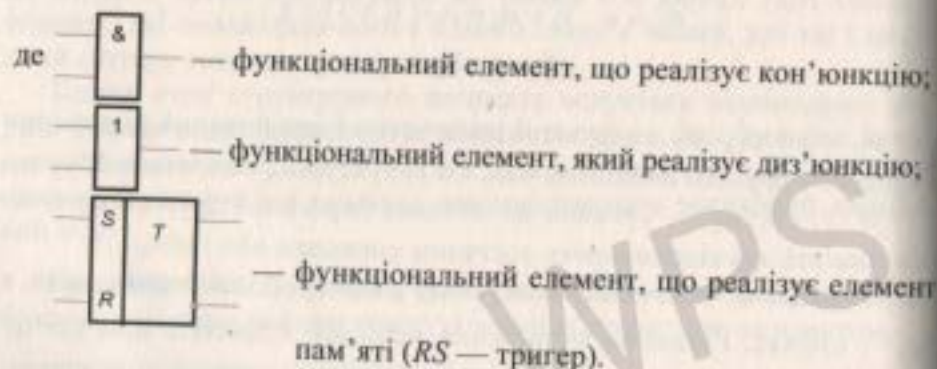


Рис. 1.4.5



Контрольні запитання

1. Що розуміють під інформаційними основами комп'ютерної техніки?
2. В яких одиницях вимірюється кількість інформації?
3. Які системи числення застосовують у комп'ютерах?
4. Яку систему числення називають двійковою, вісімковою, шістнадцятковою?
5. Як переводять цілі і дробові числа із десятикової системи числення у двійкову, вісімкову та шістнадцяткову і навпаки?
6. Що таке прямий, додатковий та обернений коди?

7. Для яких цілей застосовують додатковий і обернений коди?

8. Що таке модифікований код та для яких цілей його застосовують?

9. Що називають мантисою і порядком у двійковому числі?

10. Що називають нормалізацією числа у двійковому коді й для яких цілей її застосовують?

11. Дайте визначення логічній змінній та логічній функції?

12. Поясніть сенс поняття істинної і неістинної змінної.

13. Що називають таблицею істинності і для чого вона застосовується?

14. Що таке опис функцій алгебри логіки у вигляді алгебраїчного виразу, послідовності десятикових чисел? Де їх використовують?

15. Яку функцію алгебри логіки називають ДНФ, ДДНФ, КНФ, ДКНФ?

16. Для чого застосовується мінімізація логічних функцій?

17. Що називають автоматом? Які бувають автомати?

18. Чим відрізняються тривіальні автомати від автоматів із пам'яттю?

19. Що розуміють під абстрактним і структурним автоматом?

20. Що таке синхронний і асинхронний автомат?

21. Яка різниця між автоматом Мілі, Мура, С-автоматом?

22. Де можуть бути застосовані автомати Мілі, Мура та С-автомати?

23. Скількома способами можна задавати автомати?

24. Як визначають кількість елементів пам'яті структурного автомата, якщо відома кількість станів абстрактного автомата?

25. Опишіть графічний метод структурного синтезу автомата.



Задачі для самостійного розв'язування

1. Перевести числа 127 і 0,127 з десятикової у двійкову, вісімкову і шістнадцяткову системи числення з точністю 10^{-4} й виконати перевірку.

2. Перевести число 30,256 з десятикової у двійкову, вісімкову і шістнадцяткову системи числення з точністю 10^{-6} й виконати перевірку.

3. Виконати додавання двох двійкових чисел $A_1 = 101001$, $A_2 = 111001$.

4. Виконати віднімання двох двійкових чисел $A_1 = 110010$, $A_2 = 101101$.

5. Виконати множення і ділення двох двійкових чисел $A_1 = 11010$, $A_2 = 10101$.

6. Представити двійкові числа $A_1 = +0,10011$ і $A_2 = -0,01100$ у прямому, додатковому і оберненому кодах.

7. Виконати додавання двійкових чисел $A_1 = 0,01101$ і $A_2 = 0,10111$ у модифікованому додатковому коді за умови:

а) $A_1 > 0$; $A_2 > 0$; $A_1 + A_2 > 0$; б) $A_1 > 0$; $A_2 < 0$; $A_1 + A_2 < 0$;

в) $A_1 < 0$; $A_2 > 0$; $A_1 + A_2 > 0$; г) $A_1 < 0$; $A_2 < 0$; $A_1 + A_2 < 0$.

8. Виконати додавання двійкових чисел, наведених в задачі 7, у модифікованому оберненому коді.

9. Використовуючи плаваючу кому, виконати додавання і нормалізацію двійкових чисел $A_1 = +0,010101 \cdot 10^{+101}$ і $A_2 = -0,101101 \times 10^{+100}$.

10. Скільки логічних функцій від n -змінних приймають однакові значення на протилежних наборах? Протилежними наборами, наприклад, для двох змінних вважати набори: $\langle 0,0 \rangle - \langle 1,1 \rangle$; $\langle 1,0 \rangle - \langle 0,1 \rangle$.

11. Побудувати таблицю Карно та мінімізувати за нею наступні логічні функції:

а) $f(x, y, z) = x \cdot y \cdot z \vee \bar{x} \cdot y \cdot \bar{z} \vee x \cdot y \cdot \bar{z} \vee \bar{x} \cdot y \cdot z \vee x \cdot y \cdot z$;

б) $f(x, y, z) = x \cdot y \cdot z \vee \bar{x} \cdot y \cdot z \vee x \cdot y \cdot \bar{z} \vee \bar{x} \cdot y \cdot \bar{z} \vee x \cdot \bar{y} \cdot z$;

в) $f(x, y, z, t) = x \cdot y \cdot \bar{z} \cdot t \vee x \cdot y \cdot z \cdot t \vee \bar{x} \cdot \bar{y} \cdot \bar{z} \cdot t \vee \bar{x} \cdot y \cdot \bar{z} \cdot t \vee x \cdot y \cdot z \cdot t$;

12. За допомогою автомата Мілі, Мура або С-автомата описати системи:

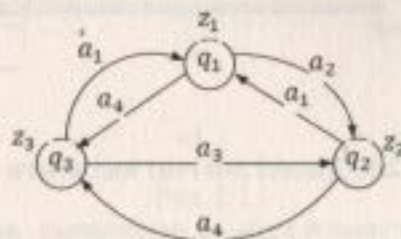
а) управління вантажним ліфтом трьохповерхового комплексу, що має кнопку виклику на кожному поверсі і працює таким чином, що: якщо натиснута одна кнопка, то ліфт рухається до поверху, де вона натиснута, якщо натиснути дві або три кнопки, то він рухається до найнижчого поверху, де натиснута кнопка.

б) накопичення числа одиниць модуля 2, що поступають до неї у вигляді двійкових сигналів 0 і 1.

13. Використовуючи графічний метод структурного синтезу автоматів розробити:

а) схему сумування модуля 3 двійкових сигналів «0» і «1»;

в) схему управління комп'ютерним пристроєм, заданою у вигляді абстрактного автомата:



Коментарі. У даному розділі інформаційні основи комп'ютерної схемотехніки викладено згідно [14], арифметичні основи комп'ютерної схемотехніки — [18], логічні основи комп'ютерної схемотехніки взяті з — [5, 7, 20], а автоматні основи комп'ютерної схемотехніки — із [4, 17].



Розділ 2

ОСНОВИ ПОБУДОВИ ЛОГІЧНИХ ЕЛЕМЕНТІВ КОМП'ЮТЕРНОЇ СХЕМОТЕХНІКИ

2.1. Діодні логічні елементи

В комп'ютерній схемотехніці, в основному, використовують потенційну систему елементів. Потенційні елементи розрізняють за схемотехнічною ознакою — способом об'єднання діодів, транзисторів і резисторів між собою в границях однієї схеми типового базового елемента. Прийнято вважати, що сукупність елементів зі спільною ознакою побудови створюють вид схемної логіки або просто логіку. Основними видами такої логіки є:

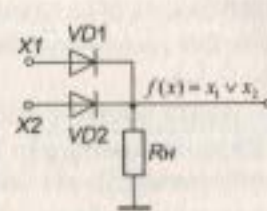
1. Діодна логіка (ДЛ);
2. Транзисторна логіка (ТЛ);
3. Діодно-транзисторна логіка (ДТЛ);
4. Транзисторно-транзисторна логіка (ТТЛ);
5. МОП — транзисторна логіка (p -МОП, n -МОП, КМОП).

Діодні логічні елементи є найпростішими схемами, які реалізують логічні функції «АБО», «І», «І-АБО», «АБО-І». Діодні логічні елементи не підсилюють вхідних сигналів і не виконують логічну операцію «НІ». Логіка роботи елемента «АБО» на два входи x_1 і x_2 представлена в табл. 2.1.1, побудований логічний елемент для реалізації функції «АБО» на двох діодах $VD1$ і $VD2$ — на рис. 2.1.1а, а часова діаграма його роботи — на рис. 2.1.1б.

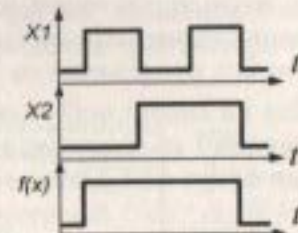
Логічна «1» на виході діодного елемента «АБО» встановлюється при подачі на один або два його входи логічної «1», при яких відкриваються відповідні діоди $VD1$ або $VD2$ або одночасно $VD1$ і $VD2$. Якщо одночасно на два діоди подати сигнал логічного «0», табл. 2.1.1, то $VD1$ і $VD2$ будуть закриті і струм у резисторі R_H не протікатиме, а тому на виході діодного логічного елемента «АБО» буде сигнал «0». Тобто дана логічна діодна схема реалізує функцію $f(x) = x_1 \vee x_2$.

Таблиця 2.1.1

x_1	x_2	$f(x)$
0	0	0
0	1	1
1	0	1
1	1	1



а)



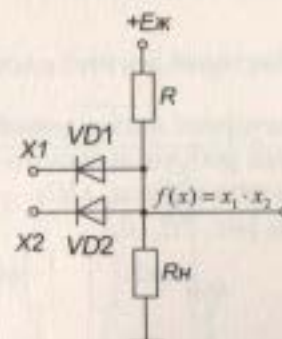
б)

Рис. 2.1.1

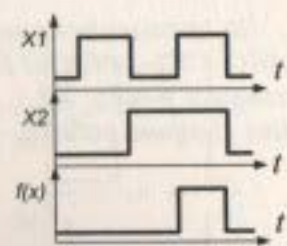
Логіка роботи діодного елемента «І» на два входи x_1 і x_2 представлена в табл. 2.1.2. Побудований логічний елемент для реалізації функції «І» на двох діодах $VD1$ і $VD2$ представлений на рис. 2.1.2а, а часова діаграма його роботи — на рис. 2.1.2б.

Таблиця 2.2.2

x_1	x_2	$f(x)$
0	0	0
0	1	0
1	0	0
1	1	1



а)



б)

Рис. 2.1.2

Логічна «1» на виході діодного елемента «І» встановлюється тільки тоді, коли одночасно будуть присутні логічні «1» на двох входах діодів $VD1$ і $VD2$. В даному випадку діоди $VD1$ і $VD2$ закриваються, і струм піде в резистор R_H , що буде відповідати логічній «1» на виході діодного логічного елемента «І», рис. 2.1.2б. Тобто дана діодна логічна схема реалізує функцію $f(x) = x_1 \cdot x_2$.

Побудова логічного діодного елемента «І – АБО» відбувається шляхом підключення, наприклад, на вхід елемента «АБО», рис. 2.1.1а, двох діодних елементів «І», рис. 2.1.2а. Схема такого елемента має вигляд, приведений на рис. 2.1.3.

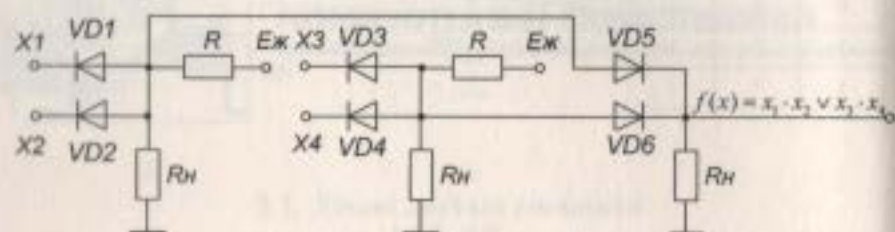


Рис. 2.1.3

Дана діодна логічна схема реалізує функцію $f(x) = x_1 \cdot x_2 \vee x_3 \cdot x_4$ і працює аналогічно описаним вище схемам при реалізації логічної функції «АБО», «І». Побудову діодного логічного елемента «АБО – І» пропонується читачеві виконати самостійно.

2.2. Транзисторні логічні елементи

За допомогою транзисторної логіки реалізують функції «НІ», «АБО – НІ», «АБО». Логіка роботи логічного елемента «НІ» представлена в табл. 2.2.1, схема елемента «НІ» — на рис. 2.2.1а, а часова діаграма роботи — на рис. 2.2.1б.

Таблиця 2.2.1

x	f(x)
0	1
1	0

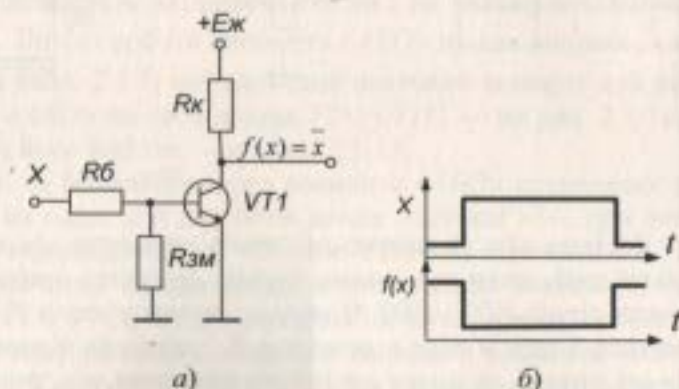


Рис. 2.2.1

Схема елемента «НІ» включає: транзистор VT1 n-p-n типу, резистори колекторного навантаження Rк, бази Rб, зміщення Rзм, джерело живлення Eж. Таку схему часто називають транзисторним ключем.

Схема елемента «НІ» працює таким чином. При подачі на вхід бази сигналу логічної «1» транзистор VT1 відкривається і переходить у стан насичення. На колекторі транзистора VT1 в цей час з'являється сигнал «0», який міститься в інтервалі $U_{ке} = 0 \dots 0,4B$. Якщо на вхід бази подається логічний сигнал «0», то транзистор VT1 закривається під дією резистора Rзм і на його колекторному виході з'являється сигнал логічної «1», що відповідає виконанню логічної функції «НІ».

Для реалізації логічної функції «АБО-НІ» в елементах транзисторної логіки «НІ» з'єднують їх колектори. Схема транзисторної логіки, яка реалізує логічну функцію «АБО-НІ» приведена на рис. 2.2.2а, часова діаграма її роботи — на рис. 2.2.2б, а функціональна схема — на рис. 2.2.2в.

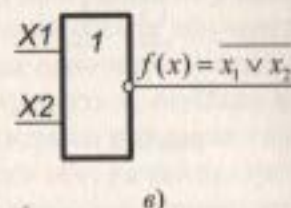
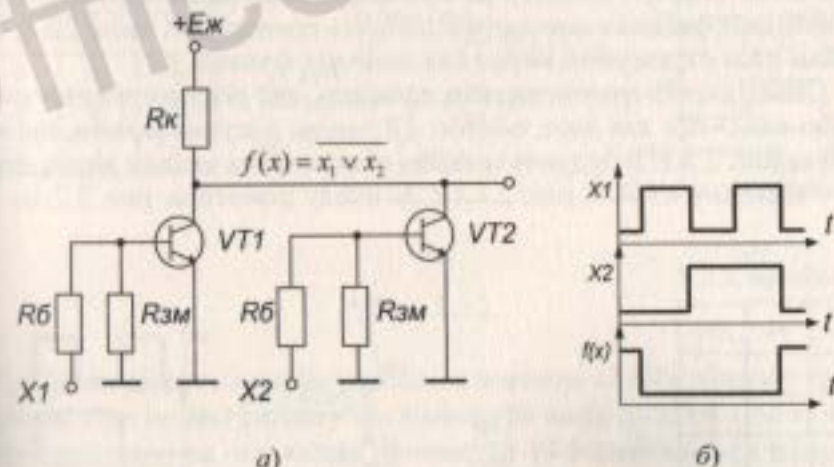


Рис. 2.2.2

Із часової діаграми випливає, що сигнал логічної «1» на виході схеми з'являється тоді і тільки тоді, коли сигнали на входах баз транзисторів $VT1$ і $VT2$ будуть одночасно дорівнювати логічному «0». На виході транзисторної схеми буде сигнал логічного «0» за наявності сигналу «1» хоча б на одному із її входів. Тобто дана схема транзисторної логіки реалізує функцію $f(x) = x_1 \vee x_2$ (стрілка Пірса).

Для реалізації логічної функції «АБО» на елементах транзисторної логіки необхідно на вихід схеми елемента «АБО-НІ», рис. 2.2.2а, підключити схему, яка реалізує елемент «НІ», рис. 2.2.1а. Накреслити таку схему пропонується читачеві.

2.3. Діодно-транзисторні логічні елементи

На елементах діодно-транзисторної логіки реалізують операції диз'юнкції і кон'юнкції за допомогою діодних схем, а операцію заперечення виконує інвертор на основі транзисторного ключа. Логіка роботи двох-вхідного елемента «АБО-НІ» приведена в табл. 2.3.1, на основі якої отримують вираз для логічної функції $f(x)$.

Схема діодно-транзисторного елемента, яка реалізує логічну операцію «АБО-НІ» для двох змінних і її часова діаграма роботи, приведені на рис. 2.3.1. Її будують шляхом підключення виходу діодів діодного елемента «АБО», рис. 2.1.1а, до входу інвертора, рис. 2.2.1а.

Таблиця 2.3.1

x_1	x_2	$f(x)$
0	0	1
0	1	0
1	0	0
1	1	0

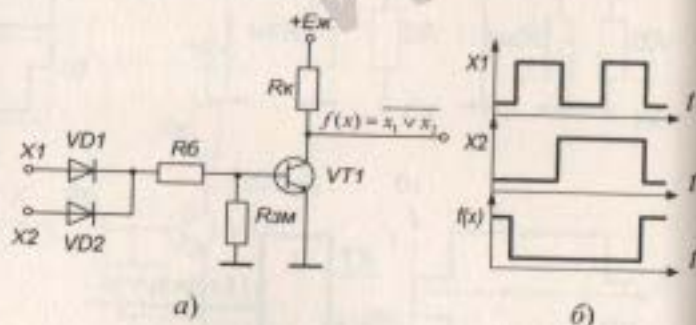


Рис. 2.3.1

Схема діодно-транзисторного елемента «АБО-НІ» працює таким чином. При подачі логічної «1» на будь-який вхід x_1 чи x_2 або

одночасно на обидва входи, транзистор $VT1$ відкривається і формує сигнал логічного «0» на своєму виході (колекторі). Якщо на входах x_1 та x_2 схеми одночасно містяться логічні сигнали «0», то під дією резистора $R_{ЗМ}$ транзистор $VT1$ закривається і на його колекторі з'являється сигнал логічної «1», на що вказує часова діаграма його роботи, рис. 2.3.1б.

Логіка роботи двох вхідних елементів «І-НІ» приведена в табл. 2.3.2, на основі якої отримують вираз для логічної функції $f(x) = x_1 \cdot x_2$. Схема діодно-транзисторного елемента, яка реалізує логічну операцію «І-НІ» для двох змінних і її часова діаграма роботи, приведені на рис. 2.3.2. Її будують шляхом підключення виходу діодного елемента «І», рис. 2.1.2а, до входу інвертора, рис. 2.2.1а.

Таблиця 2.3.2

x_1	x_2	$f(x)$
0	0	1
0	1	1
1	0	1
1	1	0

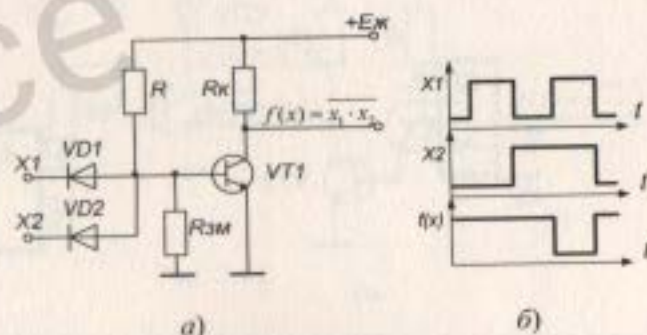


Рис. 2.3.2

Схема діодно-транзисторного елемента «І-НІ» працює таким чином. При подачі сигналу логічного «0» на будь-який вхід x_1 чи x_2 або одночасно на оба входи транзистор $VT1$ закривається і формує логічний «1» на своєму виході, так як у цей час до бази транзистора $VT1$ докладається логічний сигнал «0» через діоди $VD1$, $VD2$. Якщо на входах x_1 та x_2 схеми одночасно містяться сигнали логічної «1», то транзистор $VT1$ відкривається, оскільки до його бази через резистор R підключається джерело живлення $+E_{Ж}$, і на його виході з'являється сигнал логічного «0», на що вказує часова діаграма роботи, рис. 2.3.2б.

2.4. Транзисторно-транзисторні логічні елементи

Елементи ТТЛ з'явилися у результаті розвитку схем ДТЛ у напрямку скорочення кількості компонентів, зменшення ємності переходів і урахування специфіки інтегральної технології. Головною властивістю елементів ТТЛ було використання на вході багатоємірного транзистора (БЕТ) для реалізації операції «І», якщо входні сигнали поступають у вигляді логічних «1» і операції «АБО» у вигляді логічного «0». У ТТЛ логіці кожен емітер БЕТ використовують як логічний вхід.

Схема ТТЛ — елемента з простим інвертором на дві входні змінні x_1 і x_2 приведена на рис. 2.4.1а, а його функціональне позначення — на рис. 2.4.1б.

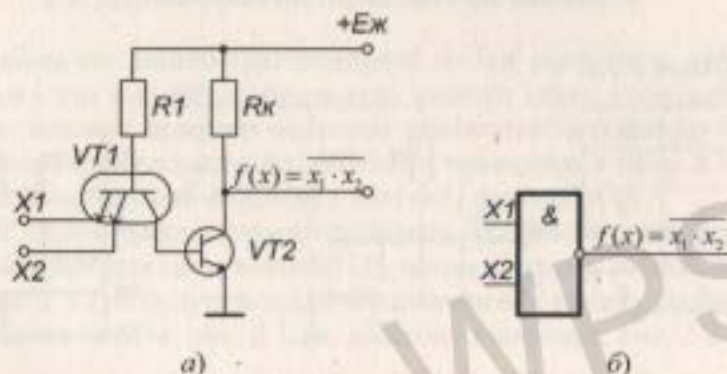


Рис. 2.4.1

Дана схема складається з БЕТ, колектор якого підключений до бази інвертуючого транзистора VT_2 . Багатоємітний транзистор VT_1 виконує логічну операцію «І», а транзистор VT_2 — операцію «НІ», тому приведений на рис. 2.4.1а елемент ТТЛ у цілому реалізує функцію $f(x_1, x_2) = x_1 \cdot x_2 = x_1 | x_2$ (функція Шеффера).

ТТЛ-елемент працює таким чином. При подачі сигналів логічної «1» на входи x_1 та x_2 емітерні переходи транзистора VT_1 закриваються, а струм його бази комутується у базу транзистора VT_2 . В результаті цього транзистор VT_2 під дією цього струму відкривається і на виході транзистора VT_2 з'являється сигнал логічного «0». Але якщо на будь-якому вході x_1 або x_2 з'явиться логічний сигнал «0», то струм бази транзистора VT_1 комутується в один із його вхо-

дів, що призведе до закриття транзистора VT_2 через відсутність у його базі струму. На виході логічного елемента ТТЛ з'явиться сигнал логічної «1».

Для збільшення швидкості і навантажувальної здатності ТТЛ-елемента використовують складні інвертори. Схема такого базового елемента приведена на рис. 2.4.2а, а його функціональне позначення — на рис. 2.4.2б.

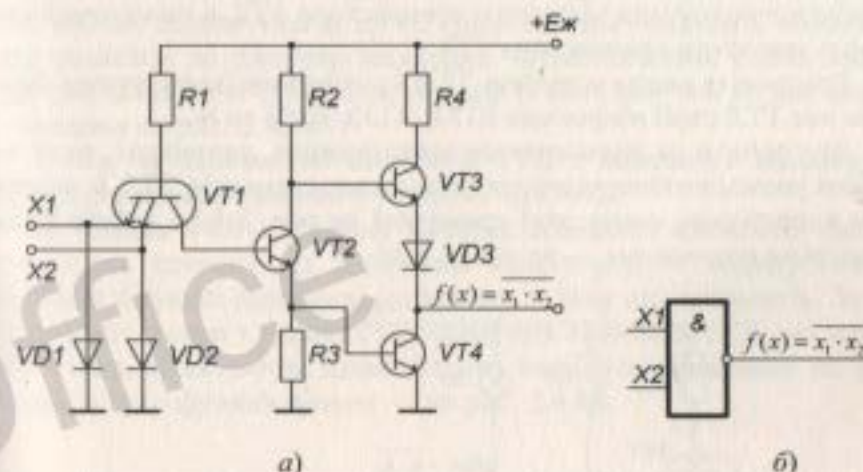


Рис. 2.4.2

Схема ТТЛ зі складним інвертором складається з трьох основних каскадів.

1. **Вхідний каскад**, який реалізує операцію «І» (багатоємітний транзистор VT_1 , резистор R_1). До всіх входів БЕТ підключені діоди VD_1 і VD_2 , які перешкоджають впливу імпульсних завад від'ємної полярності.

2. **Фазоінверсний каскад** (транзистор VT_2 , резистори R_2 і R_3), який управляє вихідними транзисторами за допомогою протифазних змін напруг на колекторі і емітері транзистора VT_2 .

3. **Вихідний двоканальний підсилювач** (транзистори VT_3 , VT_4 , з'єднаний діод VD_3 , резистор R_4). Складний інвертор утворюється спільною роботою фазоінверсного і вихідного каскадів.

Якщо на емітерні входи вхідного каскаду подати сигнали логічної «1», то частина емітерного струму транзистора VT_2 піде в базу транзистора VT_4 і відкриє його, у зв'язку з чим на його виході буде

сигнал логічного «0». При цьому транзистор $VT3$ буде закритий, так як напруги, яка прикладається до послідовного включених переходів бази і діода $VD3$ недостатньо для його відкриття.

При подачі на будь-який із входів вхідного каскаду сигналу логічного «0», струм у колекторі транзистора $VT1$ буде дорівнювати нулю, що приведе до закриття транзисторів $VT2$ і $VT4$ і відкриття транзистора $VT3$. При цьому транзистор $VT3$ працює в режимі емітерного повторювача, так як на його вхід поступає високий рівень напруги з колектора закритого транзистора $VT2$, а навантаженням є опір закритого транзистора $VT4$.

Розглянута схема елемента ТТЛ із складним інвертором є базою для ТТЛ серії мікросхем К131, К133, К155 та ін.

Для роботи на нестандартне навантаження, наприклад, реле або лампи розжарювання застосовують схеми елементів ТТЛ із відкритим колектором, схема якої приведена на рис. 2.4.3а, а його функціональне позначення — на рис. 2.4.3б.

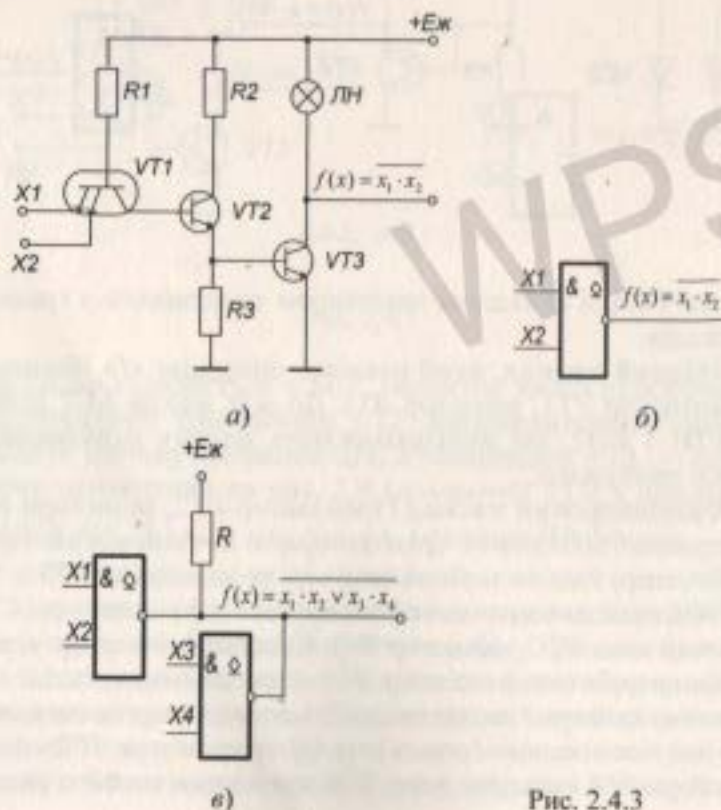


Рис. 2.4.3

Схема ТТЛ із відкритим колектором, яка приведена на рис. 2.4.3а, працює таким чином. При подачі сигналу логічної «1» на входи x_1 і x_2 схеми відкриваються транзистори $VT2$ і $VT3$, що веде до засвічення лампи розжарювання ЛН. Але якщо буде сигнал логічного «0» на будь-якому із входів x_1 або x_2 , то транзистори $VT2$ і $VT3$ закриваються, і лампа розжарювання тухне.

Виходи елементів з відкритим колектором можна об'єднувати між собою, підключивши це об'єднання через спільний колекторний резистор до джерела живлення. Функціональна схема такого використання двох елементів, кожний із яких має теж по два входи, приведена на рис. 2.4.3в.

Отже, за допомогою елементів ТТЛ з відкритим колектором можна реалізувати логічні функції «І-АБО-НІ».

В схемах ТТЛ з'єднання виходів декількох елементів (це не стосується елементів з відкритим колектором) є недопустимим, оскільки це може призвести до виходу із ладу цих елементів. За необхідності такого з'єднання застосовують елементи ТТЛ, які мають три стани. Електрична схема такого елемента приведена на рис. 2.4.4а, а його функціональна — на рис. 2.4.4б.

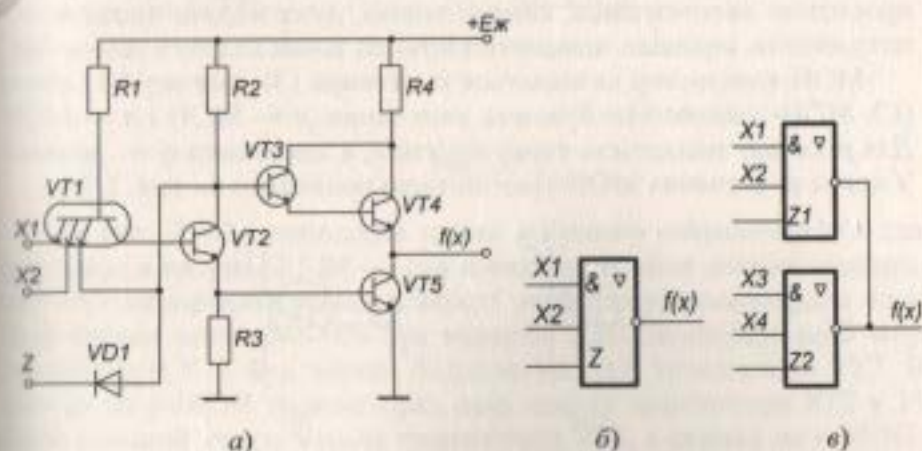


Рис. 2.4.4

Два стани виходів звичайної ТТЛ — це звичайна видача значень логічних сигналів «1» або «0». Третій стан ТТЛ характеризується безмежно великим вихідним опором, коли елемент практично

повністю відключається від навантаження, тобто не споживає і не видає струму. Ця властивість елемента може забезпечуватися рядом схемних рішень, одне із яких приведено на рис. 2.4.4а. Ця схема працює таким чином. При подачі логічного сигналу «1» на управляючий вхід Z схема ТТЛ працює як звичайний елемент «І-НІ», робота якого була описана вище для складного інвертора, рис. 2.4.2а. Якщо на управляючий вхід Z подається сигнал логічного «0», то емітер транзистора $VT1$, колектор транзистора $VT2$ і база транзистора $VT3$ підключаються через відкритий діод до логічного сигналу «0». В такому випадку всі транзистори будуть закриті, і елемент переходить у третій стан. При об'єднанні виходів елементів з трьома станами, рис. 2.4.4в, на управляючі входи $Z1$, $Z2$ сигнали повинні надходити тільки послідовно.

2.5. Логічні елементи на МОН-транзисторах

МОН-транзистори мають структуру метал — діелектрик — напівпровідник і в спільному випадку називаються МДН — транзисторами. Але через те, що в якості діелектрика в них використовують основу оксиду SiO_2 , то їх називають МОН-транзисторами.

Всі схеми на МОН-транзисторах характеризуються відносною простотою виготовлення, компактністю, дуже малою споживаною потужністю, великою завадостійкістю до зміни напруги живлення.

МОН-транзистор складається із затвора (Z), витоку (I) і стоку (C). МОН-транзистори бувають двох видів: p — МОН і n — МОН. Для p -каналу полярність стоку від'ємна, а для n -каналу — додатня. Умовне позначення МОН-транзисторів приведено на рис. 2.5.1.

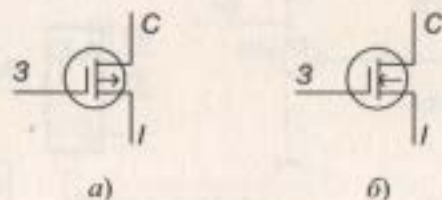


Рис. 2.5.1

На рис. 2.5.1а показано транзистор із p -МОН каналом, а на рис. 2.5.1б — із n -МОН каналом. Під каналом розуміють провідниковий прошарок між витоком і стоком, величина струму в якому

визначається за допомогою електричного поля. При нульовому значенні управляючої напруги канал у транзисторі буде відсутнім, і струм у транзисторі не протікатиме. Канал, який утворюється під дією зовнішньої управляючої напруги, називають індукованим. Напругу, при якій утворюється канал, називають пороговою U_n . Для n -МОН $U_n = + (1, 5, \dots, 2)В$, а для p -МОН $U_n = - (5, \dots, 7)В$.

Швидкодія n -МОН-транзисторів у 5–8 разів більша швидкої p -МОН транзисторів, так як рухомість електронів значно більша рухомості дірок. У МОН-схемах повністю виключені резистори, і їх роль виконують МОН-транзистори. Це дає можливість додатково спростити технологію виготовлення інтегральних мікросхем за рахунок видалення всіх пасивних елементів (резисторів).

Схеми логічних елементів «НІ» на МОН-транзисторах для різних каналів приведені на рис. 2.5.2.

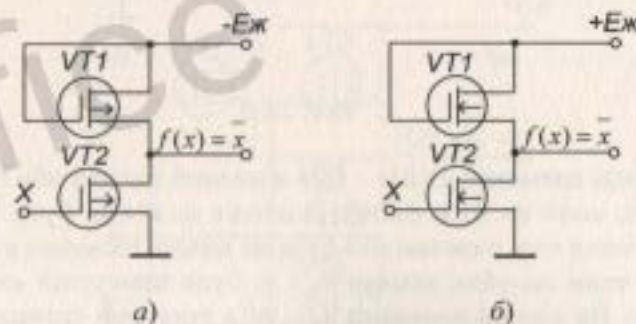


Рис. 2.5.2

На рис. 2.5.2а приведена схема логічного елемента «НІ» для p -каналу, а на рис. 2.5.2б — для n -каналу. В обох схемах транзистор $VT1$ використовується в якості навантажувального, для того щоб сигнал логічного «0» був меншим 0,1В, необхідно щоб опір транзистора $VT1$ був значно більшим за опір транзистора $VT2$. В схемах на p -МОН транзисторах опір каналу транзистора $VT1$ в 25 разів більший опору каналу транзистора $VT2$, а схемах на n -МОН-транзисторах це співвідношення близько чотирьох.

Схеми логічного елемента «НІ» працюють таким чином. Якщо на вхід схеми надходить напруга, яка менша, ніж порогова напруга транзистора, то транзистор $VT2$ закритий, а транзистор $VT1$ відкритий, і на виході схеми буде напруга близька до значення $E_{ж}$, тобто сигналу логічної «1». При подачі на вхід схеми напруги більшої за

порогову напругу (сигнал логічної «1»), то транзистор VT_2 відкривається і на його виході з'являється сигнал логічного «0».

Елементи «АБО-НІ» в схемах МОН утворюються паралельним з'єднанням входних транзисторів, рис. 2.5.3а, а елементи «І-НІ» — послідовним, рис. 2.5.3б.

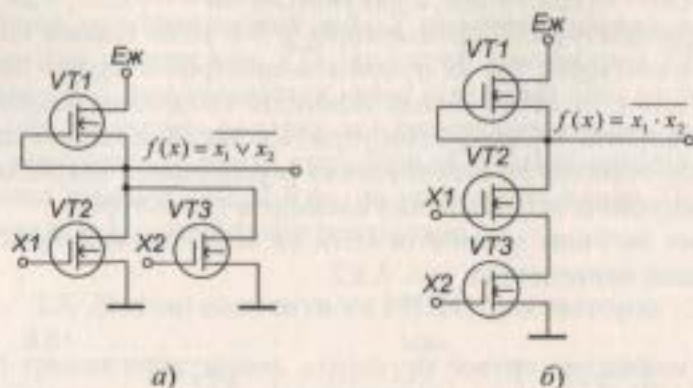


Рис. 2.5.3

На виході елемента «АБО – НІ» логічний сигнал «0» буде тоді і тільки тоді, коли на будь-якому із входів x_1 або x_2 буде присутній сигнал логічної «1», а сигнал «1» буде на виході елемента «АБО – НІ» за умови, коли на обох входах x_1 і x_2 буде присутній сигнал «0», рис. 2.5.3а. На виході елемента «І – НІ» логічний сигнал «0» буде тоді і тільки тоді, коли одночасно буде присутній сигнал «1» на входах x_1 і x_2 , рис. 2.5.3б. Тобто в цей час одночасно відкриваються транзистори VT_2 і VT_3 . Сигнал «1» на виході елемента «І – НІ» буде присутній у випадку, коли на будь-якому із його входів x_1 або x_2 буде присутній сигнал «0», це засвідчує, що один із транзисторів VT_2 або VT_3 залишиться закритим.

У комплементарній МОН-структурі в схемах одночасно використовуються n і p -канальні транзистори. Тому таку логіку називають КМОН. Схеми елементів «НІ», «АБО – НІ», «І – НІ» такої логіки приведені на рис. 2.5.4 відповідно.

Елемент «НІ», рис. 2.5.4а, побудований на двох транзисторах з індукованими каналами. Вхідний канал побудований на транзисторі VT_2 з каналом n -типу, а завантажувальний — на транзисторі VT_1 , підключеного до джерела живлення $+E_{ж}$.

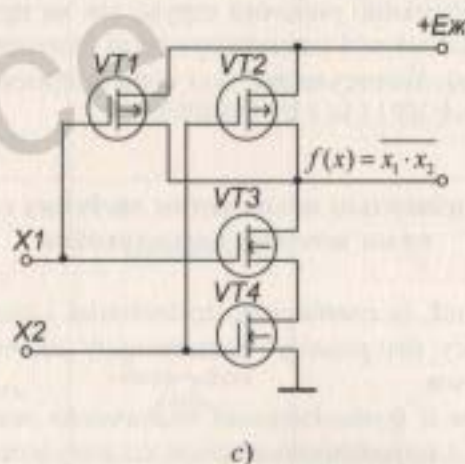
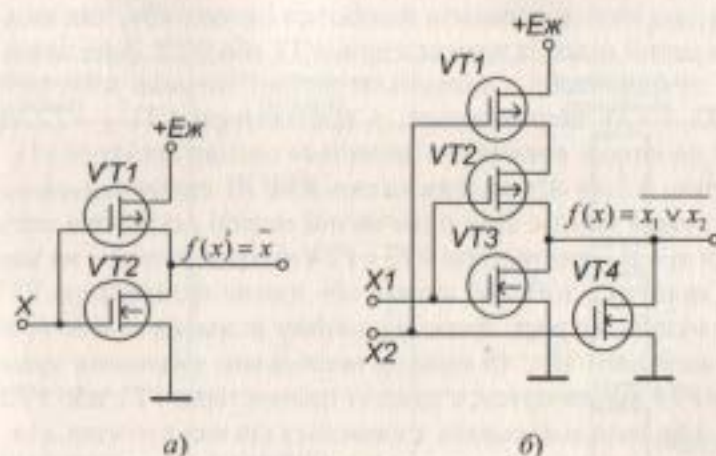


Рис. 2.5.4

Елемент «НІ» працює таким чином. При подачі на його вхід логічного сигналу «1» транзистор VT_2 відкривається, VT_1 — закривається і на виході елемента «НІ» встановлюється сигнал логічного «0». Якщо на вхід транзистора VT_2 подати сигнал логічного «0», то в елементі «НІ» відбувається все навпаки, тобто на його виході встановиться сигнал логічної «1».

На рис. 2.5.4б приведена схема КМОН елемента «АБО – НІ», яка працює таким чином. Якщо на будь-якому з входів x_1 або x_2 наявний сигнал логічної «1», то транзистори VT_3 або VT_4 відкри-

ваються і на виході елемента з'явиться сигнал «0», так як в цей час буде закритий один із транзисторів $VT1$ або $VT2$. Але якщо на входах x_1 і x_2 одночасно з'являються логічні сигнали «0», то транзистори $VT3$ і $VT4$ закриваються, а транзистори $VT1$ і $VT2$ відкриваються, і на виході елемента з'являється сигнал логічної «1».

На рис. 2.5.4в приведена схема КМОН елемента «І – НІ», яка працює таким чином. При одночасній подачі логічного сигналу «1» на входи x_1 і x_2 , транзистори $VT3$ і $VT4$ відкриваються і на виході елемента з'являється логічний сигнал «0», так як транзистори $VT1$ і $VT2$ в цей час будуть закриті. Якщо на одному із входів x_1 або x_2 з'явиться сигнал логічного «0», то один із послідовно з'єднаних транзисторів $VT3$ або $VT4$ закривається, а один із транзисторів $VT1$ або $VT2$ відкривається, і на виході елемента з'являється сигнал логічної «1».

Таким чином, в елементах схем КМОН у статичному режимі протікає дуже маленький робочий струм, так як при відкритих входних транзисторах закриті навантажувальні і навпаки.

Тепер знайшли застосування такі серії мікросхем КМОН структури: 176, 561, 564, КР1554 і КР1561.

2.6. Функціональні позначення логічних елементів комп'ютерної схемотехніки

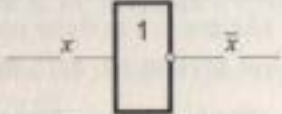
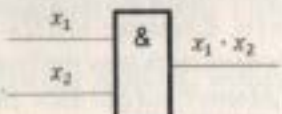
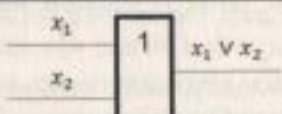
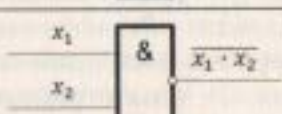
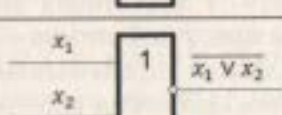
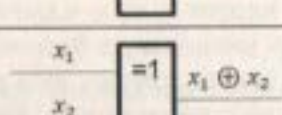
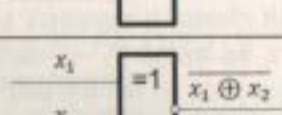
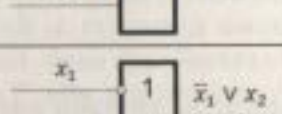
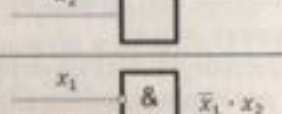
Логічні функції, їх означення, позначення і властивості приведені в § 1.3. Схему, яка реалізує елементарну логічну операцію, називають **елементом**.

Найменування й функціональні позначення основних логічних елементів згідно з нормативно-технічною документацією, наведені в табл. 2.6.1.

Наведені в табл. 2.6.1 логічні елементи є базовими при побудові різноманітних схем комп'ютерної схемотехніки. Для побудови логічних елементів, представлених у табл. 2.6.1, використовують різноманітні системи елементів, які були розглянуті в попередніх параграфах даного розділу.

Логічні елементи наведені в табл. 2.6.1 працюють наступним чином. Якщо на вхід елемента «НІ» подати логічний сигнал $x = 1$, то на його виході буде логічний сигнал «0» і навпаки. При подачі на логічний елемент «І» сигналів $x_1 = 1$ і $x_2 = 1$, на його виході буде сигнал $x_1 \cdot x_2 = 1$, а у всіх інших випадках — сигнал «0».

Таблиця 2.6.1

Найменування операції	Найменування елемента	Функціональне позначення
Заперечення	«НІ»	
Кон'юнкція	«І»	
Диз'юнкція	«АБО»	
Заперечення кон'юнкції (функція Шеффера)	«І-НІ»	
Заперечення диз'юнкції (стрілка Пірса)	«АБО-НІ»	
Нерівнозначність	«Викаюча АБО»	
Еквівалентність	«Еквівалентність»	
Імплікація	«Якщо, то»	
Заборона	«Заборона»	

Виконання операції диз'юнкції на елементі «АБО» відбувається наступним чином. При подачі сигналів $x_1 = 1$ і $x_2 = 0$ або $x_1 = 0$, $x_2 = 1$ або $x_1 = 1$, $x_2 = 1$ на його виході буде сигнал $x_1 \vee x_2 = 1$, тобто на виході елемента «АБО» в єдиному випадку буде сигнал «0», якщо на його входах x_1 і x_2 буде одночасно присутнім сигнал «0».

При подачі на вхід елемента «І-НІ» сигналів $x_1 = 1$ і $x_2 = 1$ на його виході буде сигнал $\overline{x_1 \cdot x_2} = 0$, а у всіх інших випадках — сигнал «1». Елемент «І-НІ» реалізує функцію Шеффера. Він широко застосовується при побудові інтегральних мікросхем.

Елемент «АБО-НІ» працює наступним чином. При подачі на нього хоча б одного сигнала «1» на його виході буде сигнал $x_1 \vee x_2 = 0$. Сигнал «1» на виході елемента «АБО-НІ» буде в єдиному випадку, коли на його входах x_1 і x_2 буде одночасно присутній сигнал «0». Даний елемент реалізує функцію «стрілка Пірса».

Елемент «Виключаюче АБО» виконує функцію нерівнозначності й працює наступним чином. На виході даного елемента виникає сигнал «1» тільки тоді, коли хоча б на одному вході x_1 або x_2 є сигнал «1». У всіх інших випадках на виході елемента «Виключаюче АБО» присутній сигнал «0».

Елемент «Еквівалентність» є елементом заперечення «Виключаючого АБО» і на його виході з'являється сигнал «1» тоді, коли на його входах x_1 та x_2 одночасно присутні сигнали «0» або «1». У всіх інших випадках на його виході присутній сигнал «0».

Елемент «Якщо, то» реалізує логічну функцію імплікації і працює наступним чином. На виході даного елемента сигнал буде відсутній тільки в єдиному випадку, коли на його вході x_1 буде сигнал «1», а на вході x_2 — сигнал «0». У всіх інших випадках елемент «Якщо, то» видає сигнал «1».

Елемент «Заборона» працює наступним чином. Якщо на вході даного елемента x_1 буде присутній сигнал «1», то, незалежно від значення сигналу на вході x_2 , на його виході завжди буде присутній сигнал «0». Тобто, вхід x_1 елемента «Заборона» є дозволяючим або забороняючим в залежності від значення сигналу на даному вході.

Розглянуті логічні елементи широко застосовуються для побудови логічних комбінаційних схем у комп'ютерах, таких, як дешифратори, шифратори, мультиплексори, демультиплексори, суматори та компаратори.



Контрольні запитання

1. Які логічні функції можна реалізувати на діодних і транзисторних елементах?
2. Які логічні функції можна реалізувати на діодно-транзисторних і транзисторно-транзисторних елементах?
3. Яка різниця між елементами, побудованими на МОН і КМОН-транзисторах?
4. Які із елементів, побудованих на МОН і КМОН, більше застосовуються в промисловості і чому?
5. Чим відрізняється ТТЛ-логіка від КМОН-логіки? Яка з цих логік є більш перспективною і чому?
6. Накресліть функціональні позначення елементів: «НІ»; «АБО»; «І»; «АБО — НІ»; «І — НІ»; сума за mod 2; імплікація; заборона.



Задачі для самостійного розв'язування

1. За заданими функціями $f(x, y) = x \cdot \overline{y} \vee \overline{x} \cdot y$; $f(x, y) = \overline{x} \cdot \overline{y} \vee x \cdot y$ побудувати логічні елементи, використовуючи діодну і діодно-транзисторну логіку.
2. За даними функціями $f(x, y) = \overline{x} \cdot y \vee \overline{y}$; $f(x, y) = y \vee \overline{x}$ побудувати логічні елементи, використовуючи транзисторно-транзисторну логіку.
3. За заданими функціями $f(x, y) = x \vee \overline{y}$; $f(x, y) = \overline{x \cdot y} \vee x \cdot y$ побудувати логічні елементи на МОН-транзисторах.
4. За заданими функціями прикладу 3 побудувати логічні елементи на КМОН-транзисторах.



Коментарі. В даному розділі комп'ютерної схемотехніки побудова логічних елементів із використанням діодів і транзисторів взята з [3, 18], ТТЛ-елементів впливає з [10, 24], а елементів МОН і КМОН — з [13, 18].



Розділ 3

СХЕМОТЕХНІКА ПОБУДОВИ КОМБІНАЦІЙНИХ СХЕМ

3.1. Схемотехніка побудови дешифраторів та шифраторів

Означення 3.1.1. Дешифратором називають функціональний пристрій комп'ютера, призначений для перетворення кожної комбінації вхідного двійкового коду в управляючий сигнал тільки на одному із своїх виходів.

Функціонування повного дешифратора описується такою системою логічних функцій

$$\begin{aligned} f_0 &= \bar{x}_n \cdot \bar{x}_{n-1} \cdot \dots \cdot \bar{x}_2 \cdot \bar{x}_1, \\ f_1 &= \bar{x}_n \cdot \bar{x}_{n-1} \cdot \dots \cdot \bar{x}_2 \cdot x_1, \\ &\dots \dots \dots \\ f_{m-1} &= x_n \cdot x_{n-1} \cdot \dots \cdot x_2 \cdot x_1, \end{aligned} \quad (3.1.1)$$

де $x_1, x_2, \dots, x_n$ — вхідні двійкові змінні дешифратора;

$f_0, f_1, \dots, f_{m-1}$ — вихідні логічні функції дешифратора.

Вихід дешифратора, на якому з'являється управляючий сигнал, називають активним. Якщо значення сигналу на активному виході дорівнює «1», то на всіх інших пасивних виходах дешифратора сигнали дорівнюють «0».

Активний вихід дешифратора в інтегральному виконанні часто відображається значенням логічного «0», а на решті пасивних виходах він дорівнює «1». Тоді функціонування повного дешифратора з інверсними виходами описується такою системою логічних функцій

$$\begin{aligned} \varphi_0 &= \bar{x}_n \vee \bar{x}_{n-1} \vee \dots \vee \bar{x}_2 \vee \bar{x}_1, \\ \varphi_1 &= \bar{x}_n \vee \bar{x}_{n-1} \vee \dots \vee \bar{x}_2 \vee x_1, \\ &\dots \dots \dots \\ \varphi_{m-1} &= x_n \vee x_{n-1} \vee \dots \vee x_2 \vee x_1, \end{aligned} \quad (3.1.2)$$

де $\varphi_0, \varphi_1, \dots, \varphi_{m-1}$ — вихідні логічні функції дешифратора.

Для побудови дешифратора використовують наступні кроки. На першому кроці із системи логічних функцій 3.1.1 або 3.1.2 обирають функції, які необхідно використати для проектування заданого дешифратора. На другому кроці за вибраним рівнянням у необхідній елементній базі будують потрібний дешифратор. Так, наприклад, необхідно побудувати дешифратор із чотирма активними вихідними сигналами «1» і дешифратор із чотирма активними вихідними сигналами «0». Для цього на першому кроці їх побудови складають рівняння їх роботи, які мають наступний вигляд відповідно:

$$f_0 = \bar{x}_2 \cdot \bar{x}_1; \quad f_1 = \bar{x}_2 \cdot x_1; \quad f_2 = x_2 \cdot \bar{x}_1; \quad f_3 = x_2 \cdot x_1. \quad (3.1.3)$$

$$\varphi_0 = \bar{x}_2 \vee \bar{x}_1; \quad \varphi_1 = \bar{x}_2 \vee x_1; \quad \varphi_2 = x_2 \vee \bar{x}_1; \quad \varphi_3 = x_2 \vee x_1. \quad (3.1.4)$$

На другому кроці для побудови дешифраторів використовуємо логічні елементи, які наведені в табл. 2.6.1. Схеми дешифраторів, побудовані за логічними рівняннями 3.1.3 і 3.1.4, відповідно, приведені на рис. 3.1.1а і рис. 3.1.1б.

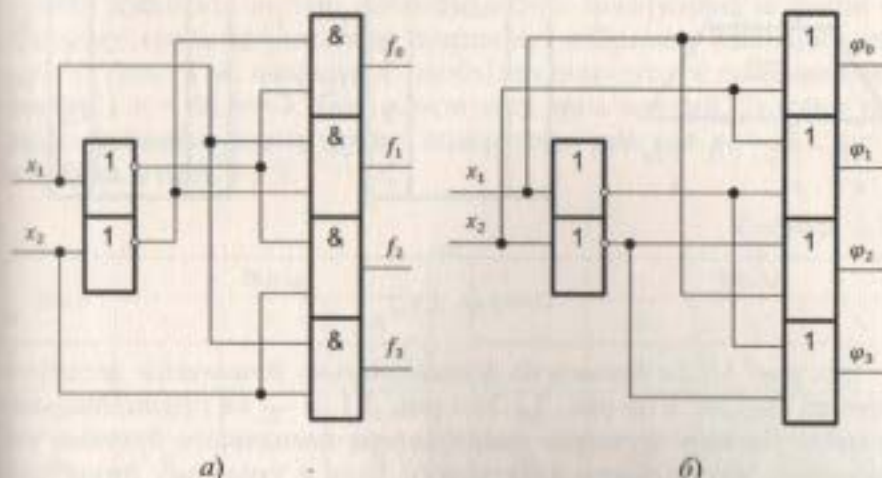


Рис. 3.1.1

На рис. 3.1.1а приведена схема дешифратора з активним вихідним сигналом «1», а на рис. 3.1.1б — з активним вихідним сигналом «0». Дані дешифратори є лінійними. Логіка їх роботи на два входи x_1 і x_2 , чотири прямих виходи (f_0, f_1, f_2, f_3) і чотири інвертних виходи ($\varphi_0, \varphi_1, \varphi_2, \varphi_3$) приведена в табл. 3.1.1 і табл. 3.1.2 відповідно.

Таблиця 3.1.1

x_2	x_1	f_0	f_1	f_2	f_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Таблиця 3.1.2

x_2	x_1	φ_0	φ_1	φ_2	φ_3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

Як видно із табл. 3.1.1 і табл. 3.1.2, логіка функціонування дешифраторів відповідає рівнянням 3.1.3 і 3.1.4, за якими будувались дані дешифратори.

Умовно графічне позначення дешифратора на електричних схемах приведено на рис. 3.1.2.

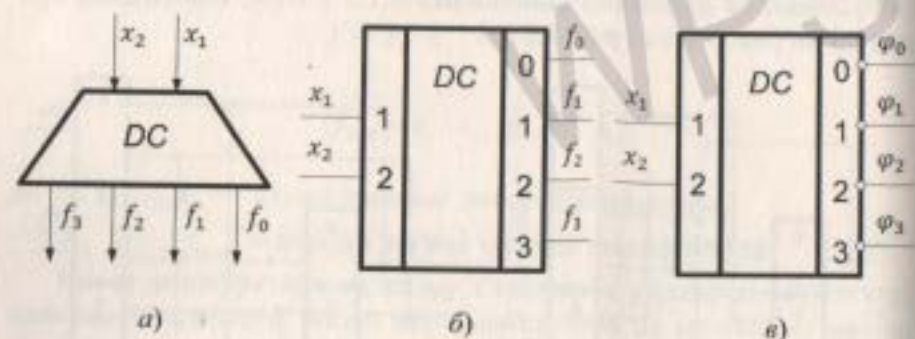


Рис. 3.1.2

На рис. 3.1.2а приведено функціональне позначення дешифратора на схемах, а на рис. 3.1.2б і рис. 3.1.2в — на принципіальних схемах. Логічну функцію дешифратора позначають буквами *DC* (decoder). Мітки лівого додаткового поля в умовному позначенні дешифратора на принципіальних схемах відтворюють десяткову

вагу вхідних змінних, а мітки правого додаткового поля відповідають десятковим еквівалентам вхідних комбінацій двійкових змінних.

В комп'ютерах дешифратори використовують для виконання таких операцій:

- а) дешифрації коду операції, записаного в регістр команд процесора, що забезпечує вибір необхідної мікропрограми;
- б) перетворення коду адреси операнди команди в управляючі сигнали вибору заданої комірки пам'яті в процесі запису або читання інформації;
- в) забезпечення візуалізації на зовнішніх пристроях;
- г) реалізації логічних операцій, побудови мультиплексорів і демультиплексорів.

Означення 3.1.2. Шифратором називають функціональний пристрій комп'ютера, призначений для перетворення вхідного m -розрядного унітарного позиційного коду у вихідний n -розрядний двійковий позиційний код.

Двійкові шифратори виконують функцію, обернену функції дешифратора. При активізації одного із входів шифратора на його виходах формується код, який відображає номер активного входу. Повний двійковий шифратор має $m = 2^n$ входів і n -виходів.

Для побудови шифраторів використовують три кроки. На першому кроці за допомогою таблиці функціонування описують роботу необхідного шифратора. На другому кроці, користуючись таблицею, знаходять логічні рівняння роботи шифратора, за якими на третьому кроці будують сам шифратор, вибравши необхідну елементну базу. Так, наприклад, необхідно побудувати двійковий шифратор з $m = 8$ і $n = 3$. Тоді, виконуючи крок перший, будемо таблицю функціонування такого шифратора, яка для $m = 8$ і $n = 3$ наведена в табл. 3.1.3.

Таблиця 3.1.3

Вхід	Вихід		
	x_3	x_2	x_1
φ_0	0	0	0
φ_1	0	0	1
φ_2	0	1	0
φ_3	0	1	1

На другому кроці, користуючись таблицею функціонування шифратора, знаходимо логічні рівняння його роботи:

$$\begin{aligned} x_1 &= \varphi_1 \vee \varphi_3 \vee \varphi_5 \vee \varphi_7; & x_2 &= \varphi_2 \vee \varphi_3 \vee \varphi_6 \vee \varphi_7; \\ x_3 &= \varphi_4 \vee \varphi_5 \vee \varphi_6 \vee \varphi_7. \end{aligned} \quad (3.1.5)$$

Для реалізації рівнянь 3.1.5 на елементній базі «І-НІ», яка широко застосовується в периферійних пристроях комп'ютерів, перетворюємо їх за законами алгебри логіки на такий вид:

$$\begin{aligned} x_1 &= \overline{\varphi_1} \cdot \overline{\varphi_3} \cdot \overline{\varphi_5} \cdot \overline{\varphi_7}; & x_2 &= \overline{\varphi_2} \cdot \overline{\varphi_3} \cdot \overline{\varphi_6} \cdot \overline{\varphi_7}; \\ x_3 &= \overline{\varphi_4} \cdot \overline{\varphi_5} \cdot \overline{\varphi_6} \cdot \overline{\varphi_7}. \end{aligned} \quad (3.1.6)$$

На третьому кроці, використовуючи рівняння 3.1.6, будемо сам шифратор, наведений на рис. 3.1.3.

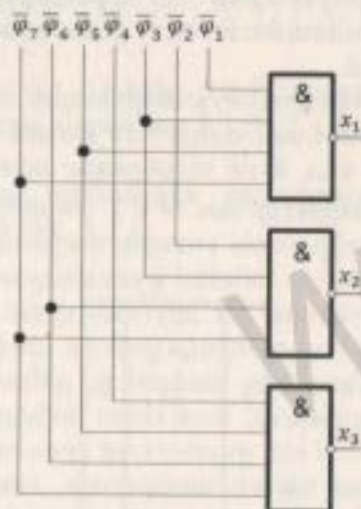


Рис. 3.1.3

Умовно графічне позначення шифраторів на схемах наведені на рис. 3.1.4.

На рис. 3.1.4а приведено функціональне позначення шифратора, а на рис. 3.1.4б — принципальне. Логічну функцію шифратора позначають буквами *CD* (*coder*). Входи шифратора нумерують цифрами (0, 1, 2, ..., $m-1$), а мітки виходів відтворюють вагу вихідних двійкових змінних (1, 2, 4, ..., 2^{n-1}).

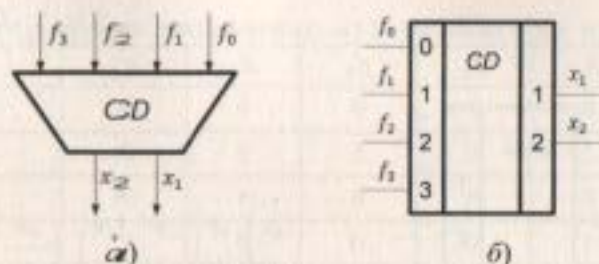


Рис. 3.1.4

В комп'ютерах шифратори використовують для виконання наступних функцій:

- перетворення унітарного вхідного коду у вихідний двійковий позиційний код;
- вводу десяткових даних з клавіатури комп'ютера;
- показу старшої одиниці в слові;
- передачі інформації між різними пристроями при обмеженні на кількість ліній зв'язку.

3.2. Схемотехніка побудови мультиплексорів і демультиплексорів

Означення 3.2.1. Мультиплексором називають функціональний пристрій комп'ютера, призначений для послідовної комутації інформації від одного із n входів на його спільний вихід.

Входи мультиплексора поділяють на інформаційні й керуючі (адресні). Конкретний вхід мультиплексора, що підключається до його виходу, визначається адресним кодом $A_0, A_1, \dots, A_{n-1}$. Зв'язок між числом інформаційних n і адресних m ходів впливає із співвідношення $n = 2^m$.

Для побудови мультиплексорів використовують три кроки. На першому кроці за допомогою таблиці функціонування описують логіку роботи мультиплексора. На другому кроці, користуючись таблицею функціонування, визначають вихідну функцію роботи мультиплексора, за якою на третьому кроці, вибравши елементну базу, будують необхідний мультиплексор.

Так, наприклад, необхідно побудувати мультиплексор, який має чотири входи. Тоді, згідно з першим кроком побудови, користуючись логікою роботи мультиплексора, будемо таблицю його функціонування, табл. 3.2.1.

Таблиця 3.2.1

A_1	A_0	F_0	F_1	F_2	F_3	D
0	0	1	0	0	0	$F_0 \cdot x_0$
0	1	0	1	0	0	$F_1 \cdot x_1$
1	0	0	0	1	0	$F_2 \cdot x_2$
1	1	0	0	0	1	$F_3 \cdot x_3$

В таблиці прийняті такі позначення: A_1, A_0 — адресні входи мультимплексора; F_0, F_1, F_2 і F_3 — виходи внутрішнього дешифратора; x_0, x_1, x_2, x_3 — вхідна інформація мультимплексора; D — спільний інформаційний вихід мультимплексора.

На другому кроці, користуючись таблицею функціонування мультимплексора, визначимо його вихідну функцію D .

$$D = F_0 \cdot x_0 \vee F_1 \cdot x_1 \vee F_2 \cdot x_2 \vee F_3 \cdot x_3. \quad (3.2.1)$$

У формулі 3.2.1 використані виходи $F_0 \dots F_3$ внутрішнього дешифратора мультимплексора. Використовуючи табл. 3.2.1, формулу 3.2.1 можна також записати його вихідну функцію D із застосуванням змінних адресного входу

$$D = \bar{A}_1 \cdot \bar{A}_0 \cdot x_0 \vee \bar{A}_1 \cdot A_0 \cdot x_1 \vee A_1 \cdot \bar{A}_0 \cdot x_2 \vee A_1 \cdot A_0 \cdot x_3. \quad (3.2.2)$$

На третьому кроці, використовуючи логічні рівняння 3.2.1 і 3.2.2, будемо необхідні мультимплексори на елементах «І», «НІ», «АБО», рис. 3.2.1.

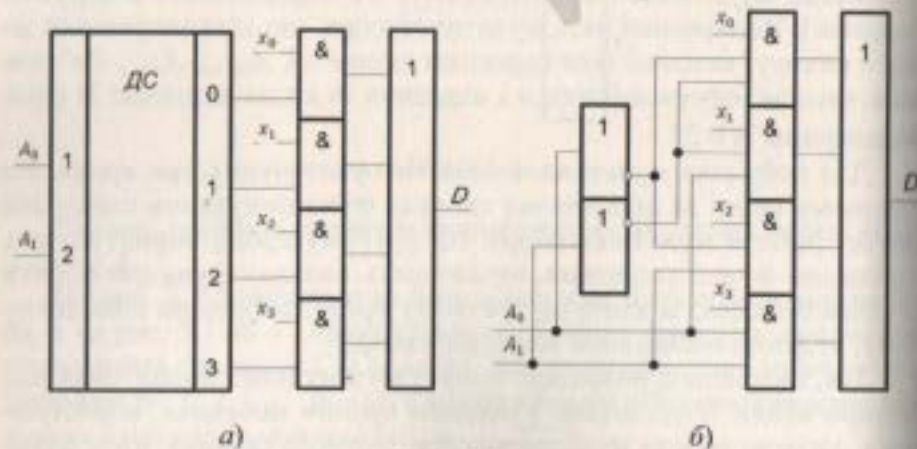


Рис. 3.2.1

Умовно графічне позначення мультимплексорів приведено на рис. 3.2.2.

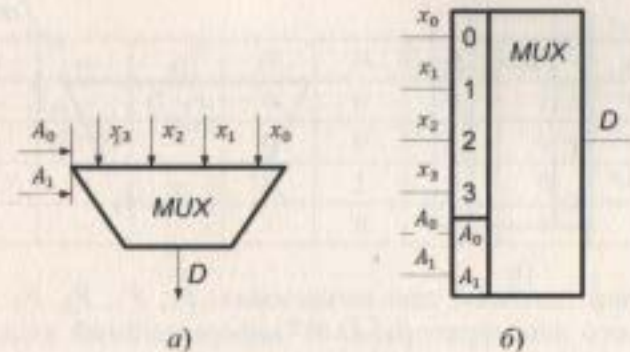


Рис. 3.2.2

На рис. 3.2.2а приведено позначення мультимплексора на функціональних, а на рис. 3.2.2б — принципальних схемах. Функцію мультимплексора позначають буквами *MUX* (*multiplexor*).

Мультимплексори в комп'ютерах використовують для виконання наступних операцій:

- комутації як окремих ліній, так і шин передачі інформації;
- перетворення паралельного коду в послідовний;
- реалізації логічних функцій.

Означення 3.2.2. Демультимплексором називають функціональний пристрій комп'ютера, призначений для комутації сигналу з одного інформаційного входу D на один із n інформаційних виходів.

Входи демультимплексора, також, як і в мультимплексорі, поділяють на інформаційні та керуючі. Номер виходу, на який у кожний момент часу передається значення вхідного сигналу, визначається адресним кодом $A_0, A_1, \dots, A_{n-1}$. Адресні входи m та інформаційні виходи n пов'язані співвідношенням $n = 2^m$.

Для побудови демультимплексорів використовують також три кроки. На першому кроці за допомогою таблиць функціонування описують логіку роботи демультимплексора. На другому, користуючись таблицею функціонування, визначають вихідні функції роботи демультимплексора, за яким на третьому кроці, вибравши елементну базу, будують необхідний демультимплексор.

Так, наприклад, необхідно побудувати демультимплексор, який має чотири входи. Тоді, згідно з першим кроком, користуючись логі-

кою роботи демультимплексора, будемо таблицю його функціонування, табл. 3.2.2.

Таблиця 3.2.2

A_1	A_0	F_0	F_1	F_2	F_3	x_0	x_1	x_2	x_3
0	0	1	0	0	0	$F_0 \cdot D$	—	—	—
0	1	0	1	0	0	—	$F_1 \cdot D$	—	—
1	0	0	0	1	0	—	—	$F_2 \cdot D$	—
1	1	1	0	0	1	—	—	—	$F_3 \cdot D$

В таблиці прийняті такі позначення: F_0, F_1, F_2, F_3 — виходи внутрішнього дешифратора; D — інформаційний вхід; x_0, x_1, x_2, x_3 — інформаційні виходи демультимплексора.

На другому кроці, користуючись таблицею функціонування демультимплексора, визначимо його вихідні функції x_0, x_1, x_2, x_3 :

$$\begin{aligned} x_0 &= F_0 \cdot D = \bar{A}_1 \cdot \bar{A}_0 \cdot D; & x_1 &= F_1 \cdot D = \bar{A}_1 \cdot A_0 \cdot D; \\ x_2 &= F_2 \cdot D = A_1 \cdot \bar{A}_0 \cdot D; & x_3 &= F_3 \cdot D = A_1 \cdot A_0 \cdot D. \end{aligned} \quad (3.2.3)$$

На третьому кроці на базі рівнянь 3.2.3 будемо схеми демультимплексорів із внутрішнім дешифратором рис. 3.2.3а і з адресними входами змінних на трьох входних елементах «І», рис. 3.2.3б.

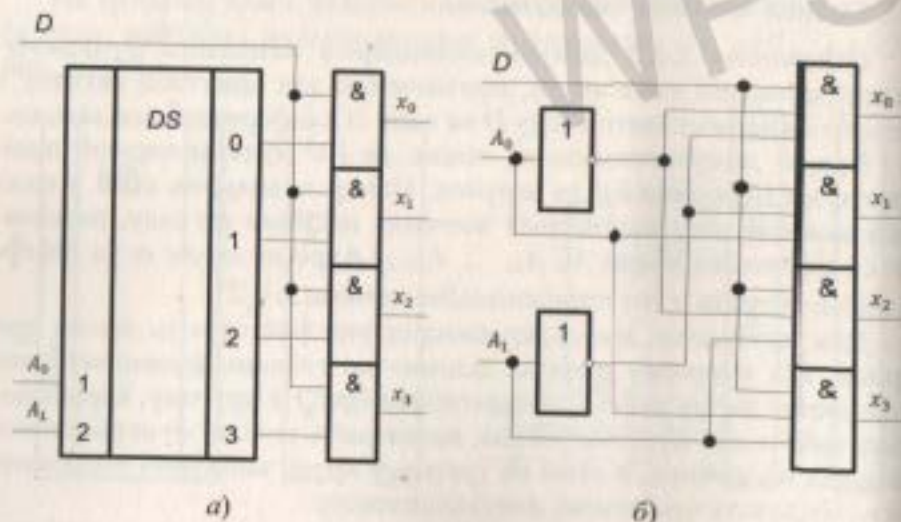


Рис. 3.2.3

Умовно графічне зображення демультимплексорів приведено на рис. 3.2.4.

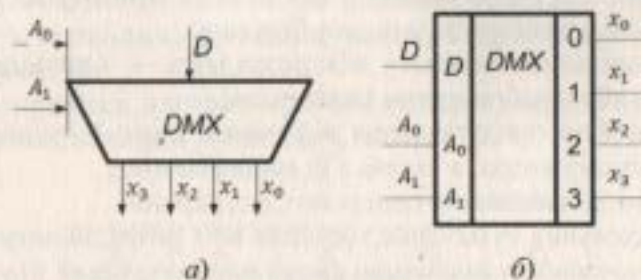


Рис. 3.2.4

На рис. 3.2.4а приведено позначення демультимплексора на функціональних, а на рис. 3.2.4б — принципальних схемах. Функцію демультимплексора позначають буквами *DMX* (*demultiplexor*).

Демультимплексор використовують для виконання таких операцій:

- комутації як окремих ліній, так і шин передачі інформації в комп'ютерах;
- перетворення послідовного коду в паралельний;
- реалізації логічних функцій.

3.3. Схемотехніка побудови суматорів

Означення 3.3.1. Суматором називають комбінаційний логічний пристрій комп'ютера, призначений для виконання операцій арифметичного додавання чисел, поданих у вигляді двійкових кодів.

Операція віднімання в суматорі замінюється операцією додавання двійкових чисел у додатковому або оберненому коді, §1.2. Операція множення і ділення зводиться до багатократного додавання і зсуву. Тому суматор є важливою частиною арифметично-логічного пристрою комп'ютера. Функцію суматора позначають буквами *SM* або Σ .

Суматор складається з окремих схем, які називаються **однорозрядними суматорами**. Ці схеми виконують усі дії з додавання значень однойменних розрядів двійкових чисел. Суматори класифікують за такими ознаками:

- способом додавання — паралельні, послідовні, паралельно-послідовні;

б) числом входів — напівсуматори, однорозрядні і багаторозрядні суматори;

в) організацією переносу зберігання результату додавання — комбінаційні, накопичувальні, комбіновані;

г) організацією переносу між розрядами — з послідовним, паралельним або комбінованим переносом;

д) способом представлення від'ємних чисел — у додатковому або оберненому кодах, а також в їх модифікаціях;

е) часом додавання — синхронні, асинхронні.

В паралельних суматорах значення всіх розрядів операндів надходять одночасно на відповідні входи однорозрядних підсумованих схем. В послідовних суматорах значення розрядів операндів і перенос, який запам'ятовується в попередньому такті, надходить послідовно в напрямку від молодших розрядів до старших на входи одного однорозрядного суматора. В паралельно-послідовних суматорах числа дробляться на частини, наприклад, байти, розряди байтів надходять на входи восьмирозрядного суматора паралельно (одночасно), а самі байти — послідовно, в напрямку від молодших до старших з урахуванням запам'ятовуючого переносу.

В комбінаційних суматорах результат операції додавання запам'ятовується в регістрі результату. В накопичувальних суматорах процес додавання об'єднується зі зберіганням результату, що пояснюється використанням Т-тригерів як однорозрядних схем додавання.

Організація переносу в суматорі практично визначає час виконання операції додавання. Послідовні переноси схемно утворюються просто, але вони не є швидкодієвими. Паралельні переноси схемно значно складніші, але значно швидкодієвими.

Суматори, які мають постійний інтервал часу для додавання називають **синхронними**, а суматори, в яких додавання визначається моментом фактичного закінчення операції, — **асинхронними**.

Означення 3.3.2. Однорозрядним суматором називають логічну схему, яка виконує додавання значень i -х розрядів x_i і y_i двійкових чисел з урахуванням переносу z_i із молодшого сусіднього розряду і дає на виходах функції результат S_i і перенос P_i в старший сусідній розряд.

На основі однорозрядних схем додавання на три входи і два виходи будуються багаторозрядні суматори будь-якого типу. Логіка побудови суматорів аналогічна логіці побудови комбінацій-

них схем, яка розглянута у попередніх розділах. Тобто, спочатку визначають алгоритм функціонування, який звичайно, подають таблицею, а потім за допомогою рівняння (системи рівнянь), які отримують із таблиці функціонування, будують необхідний за умовою суматор.

Так, наприклад, алгоритм роботи однорозрядного суматора відображається таблицею істинності, табл. 3.3.1.

Таблиця 3.3.1

x_i	y_i	z_i	S_i	P_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Із таблиці 3.3.1 випливає система логічних функцій для результату S_i і переносу в P_i в ДДНФ.

$$S_i = \bar{x}_i \cdot \bar{y}_i \cdot z_i \vee \bar{x}_i \cdot y_i \cdot \bar{z}_i \vee x_i \cdot \bar{y}_i \cdot \bar{z}_i \vee x_i \cdot z_i \cdot y_i, \quad (3.3.1)$$

$$P_i = \bar{x}_i \cdot y_i \cdot z_i \vee x_i \cdot \bar{y}_i \cdot z_i \vee x_i \cdot y_i \cdot \bar{z}_i \vee x_i \cdot y_i \cdot z_i. \quad (3.3.2)$$

Мінімізація функцій 3.3.1 і 3.3.2 за допомогою карт Карно приведена на рис. 3.3.1а і рис. 3.3.1б, відповідно

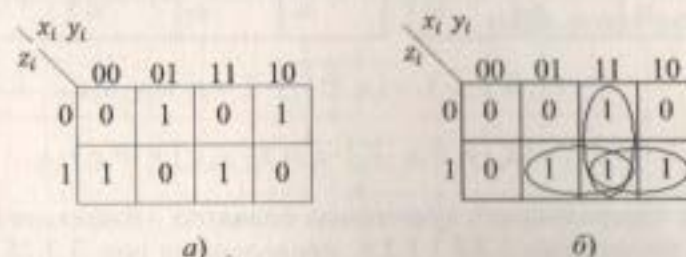


Рис. 3.3.1

Як випливає із карт Карно, функція S_i не мінімізується, а функція P_i мінімізується зі зменшенням рангу кон'юнкції і використовує тільки прямі значення змінних

$$P_i = x_i \cdot y_i \vee x_i \cdot z_i \vee y_i \cdot z_i = x_i \cdot y_i \vee (x_i \vee y_i) \cdot z_i. \quad (3.3.3)$$

При проектуванні комбінаційних однорозрядних суматорів необхідно враховувати такі фактори:

а) схема повинна мати однаковість структури і мінімальну вартість, тобто мати по можливості мінімальну кількість входів у всіх елементів;

б) для швидкодії багаторозрядного суматора необхідний мінімальний час отримання функції переносу $t_n = k \cdot t_p$, де k — кількість послідовно включених елементів від входів до виходів P_i або $\bar{P}_i$, t_p — середня затримка розповсюдження сигналу одним логічним елементом;

в) для схем однорозрядних суматорів на основі рівнянь 3.3.1 і 3.3.2 необхідно проектувати як прямі P_i , так інверсні $\bar{P}_i$ значення функції переносу.

Для побудови схеми однорозрядного суматора на логічних елементах «І-НІ», рівняння 3.3.1 і 3.3.3, з використанням подвійної інверсії і правил де-Моргана, перетворюємо на такий вигляд

$$S_i = \overline{\overline{x_i \cdot y_i \cdot z_i} \cdot \overline{x_i \cdot y_i \cdot \bar{z}_i} \cdot \overline{x_i \cdot \bar{y}_i \cdot \bar{z}_i} \cdot \overline{\bar{x}_i \cdot y_i \cdot z_i}},$$

$$P_i = \overline{x_i \cdot y_i \cdot x_i \cdot z_i \cdot y_i \cdot z_i}. \quad (3.3.4)$$

Схема однорозрядного суматора, побудована на елементах «І-НІ» у відповідності з рівняннями 3.3.4, приведена на рис. 3.3.2а.

Рівняння 3.3.1 і 3.3.2 можуть бути також виражені через функцію «Виключаюче АБО»

$$S_i = (x_i \oplus y_i) \cdot \bar{z}_i \vee (x_i \oplus y_i) \cdot z_i = x_i \oplus y_i \oplus z_i, \quad (3.3.5)$$

$$P_i = x_i \cdot y_i \vee (\bar{x}_i \cdot y_i \vee x_i \cdot \bar{y}_i) \cdot z_i = x_i \cdot y_i \vee (x_i \oplus y_i) \cdot z_i. \quad (3.3.6)$$

Схема однорозрядного суматора на елементах «Виключаюче АБО», згідно із рівняннями 3.3.5 і 3.3.6, приведена на рис. 3.3.2б, а його функціональна схема на рис. 3.3.2в.

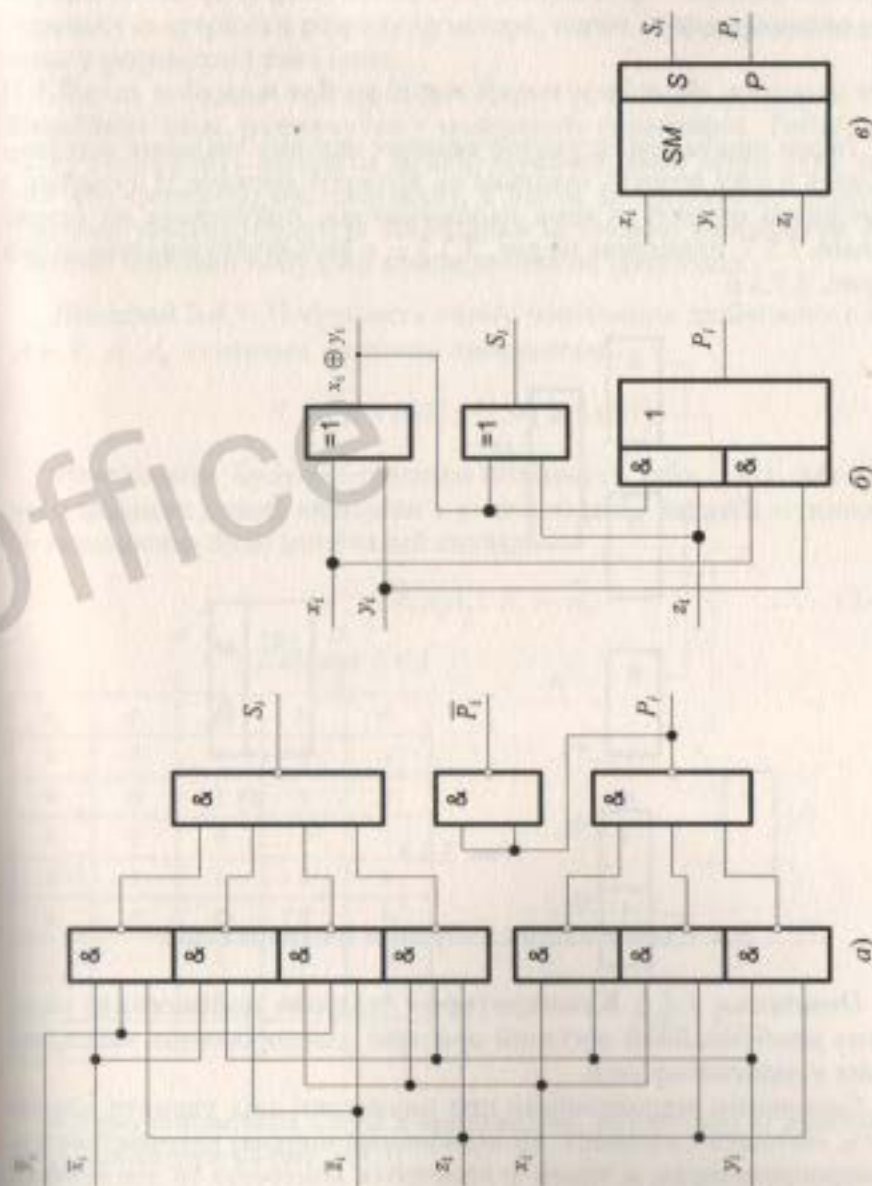


Рис. 3.3.2

Означення 3.3.3. Напівсуматором називають логічну схему, яка виконує додавання значень i -х розрядів x_i і y_i двійкових чисел x , y , і реалізує на виході значення результату M_i і перенос в старший сусідній розряд R_i .

$$M_i = \bar{x}_i \cdot y_i \vee x_i \cdot \bar{y}_i = x_i \oplus y_i; R_i = x_i \cdot y_i. \quad (3.3.7)$$

Таким чином, напівсуматор виконує частину завдання підсумовування в i -му розряді, оскільки не враховує перенос із сусіднього молодшого розряду. Схема напівсуматора, побудована на основі рівнянь 3.3.7, приведена на рис. 3.3.3 а, а його функціональна схема на рис. 3.3.3 б.

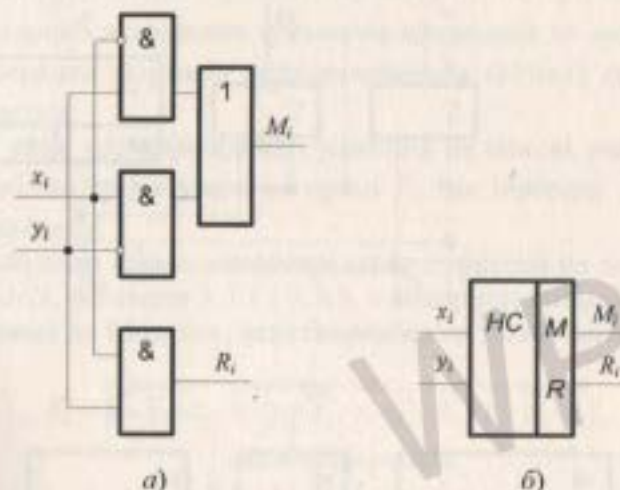


Рис. 3.3.3

3.4. Схемотехніка побудови компараторів

Означення 3.4.1. Компаратором (схемою порівняння) називають комбінаційний логічний пристрій для порівняння чисел, поданих у двійковому коді.

Основними відношеннями при порівнянні слід уважати «дорівнює», «більше» і «менше». Ці відношення широко використовують у мікропроцесорах, а також у пристроях контролю та діагностики комп'ютерів. Після виконання конкретної команди в комп'ютері автоматично формулюються ознаки результатів операції. Ці ознаки

називають **прапорці** і їх розміщують у спеціальний регістр прапорів. До прапорців, наприклад, відносять ознаки нульового результату, переповнення розрядної сітки, знак результату операції, наявність переносу із старшого розряду суматора, парне і непарне число одиниць у результаті і таке інше.

Логіка побудови компараторів аналогічна логіці побудови комбінаційних схем, розглянутих у попередніх параграфах. Тобто, спочатку визначають алгоритм функціонування, який може бути заданий або словесно, або таблицею, а потім за допомогою рівняння (системи рівнянь) будують необхідний за умовою компаратор. Розглянемо принцип побудови компараторів на прикладах.

Приклад 3.4.1. Побудувати схему порівняння двійкового слова $A = A_2 A_1 A_0$ із такими заданими константами:

$$F_1 := (A = 001); F_2 := (A \leq 001).$$

Розв'язання. Будуємо таблицю істинності, табл. 3.4.1, для заданого вхідного слова, виходячи з умови задачі і таблиці істинності, це відношення буде мати такий вигляд:

$$F_1 := \bar{A}_2 \bar{A}_1 A_0; F_2 := \bar{A}_2. \quad (3.4.1)$$

Таблиця 3.4.1

A_2	A_1	A_0	F_1	F_2
0	0	0	0	1
0	0	1	1	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	0	0

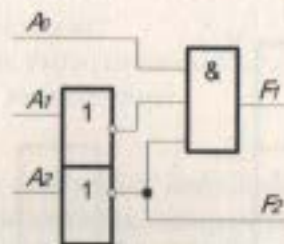


Рис. 3.4.1

Схема порівняння слова з константою, відповідно із рівнянням 3.4.1, приведена на рис. 3.4.1.

Приклад 3.4.2. Побудувати схему порівняння двох i -х розрядів $A_i = B_i$. Для порівняння будемо таблицю істинності, табл. 3.4.2.

Таблиця 3.4.2

A_i	B_i	r_i
0	0	1
0	1	0
1	0	0
1	1	1

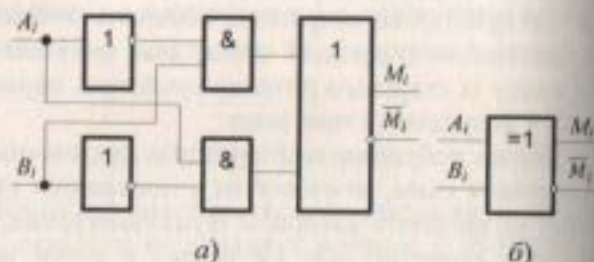


Рис. 3.4.2

Із таблиці істинності, яка задає умову рівності M_i двох i -х розрядів A_i і B_i отримаємо

$$r_i = A_i \cdot B_i \vee \bar{A}_i \cdot \bar{B}_i = \overline{A_i \oplus B_i} = \bar{M}_i, \quad (3.4.2)$$

де M_i — функція додавання за модулем два ($\text{mod}2$ — «Виключаюче АБО»).

Принципіальна схема, яка реалізує цю функцію, показана на рис. 3.4.2а, а її функціональна схема приведена на рис. 3.4.2б.

Приклад 3.4.3. Побудувати схему порівняння двох чотирьохрозрядних двійкових слів A та B .

Розв'язання. Ознака рівності двох чотирьохрозрядних слів A та B визначається добутком порозрядних умов r_i



Рис. 3.4.3



Контрольні запитання

1. Що називають дешифратором?
2. Що називають шифратором?
3. Яка різниця між дешифратором і шифратором?
4. Де використовують дешифратор і шифратор?
5. Як позначають на схемах дешифратори і шифратори?
6. Накресліть функціональну і принципіальну схеми дешифратора і шифратора.
7. Що називають мультиплексором?
8. Що називають демultipлексором?
9. Яка різниця між мультиплексором і демultipлексором?
10. Як позначають на схемах демultipлексор і мультиплексор?
11. Накресліть функціональну і принципіальну схеми демultipлексора і мультиплексора.
12. Де використовують демultipлексор і мультиплексор?
13. Що називають суматором?
14. Які операції виконує суматор?
15. Як позначають суматори на схемах?
16. Які бувають суматори?
17. За якими ознаками класифікують суматори?
18. Що називають напівсуматором і чим він відрізняється від суматора?
19. Як позначають напівсуматор на схемах?
20. Що називають компаратором і як його позначають?
21. Для яких цілей використовують компаратор?



Коментарі. В даному розділі для логіки побудови дешифраторів, шифраторів, мультиплексорів, демultipлексорів використані матеріали літератури [25, 27], а логіка побудови суматорів і компараторів випливає з [3, 17, 27].



Розділ 4

СХЕМОТЕХНІКА ПОБУДОВИ КОМБІНАЦІЙНИХ СХЕМ НА ПРОГРАМОВАНИХ ЛОГІЧНИХ МАТРИЦЯХ

4.1. Призначення і ділянки застосування

Програмовані логічні матриці (ПЛМ) являють собою логічну схему для перетворення множини вхідних значень $X = \{x_1, x_2, \dots, x_m\}$ у відповідну множину вихідних даних $Y = \{y_1, y_2, \dots, y_n\}$ в двійковому коді. Правило перетворення вхідних змінних у функціях задається таблицею істинності. ПЛМ реалізує систему булевих функцій, представлених у ДДНФ або МДНФ, або в ДНФ.

Програмовані логічні матриці знайшли широке застосування в логічних інтегральних схемах (ПЛІС). Наприклад, ПЛІС із плавкими запобіжниками за технологією ТТЛШ, які виготовляються в НДУМЕ, м. Зеленоград, Росія. В їх складі уже давно відомі ПЛМ K556PT1, KР556PT2, KР556PT21.

Завжди виникає питання, де можливо застосовувати ПЛІС?

По-перше, при розробці оригінальних і нестандартних пристроїв у комп'ютерах та системах управління, а також для заміни звичайних інтегральних мікросхем малого і середнього ступеня інтеграції. При цьому значно зменшуються розміри, потужність споживання й збільшується надійність пристроїв і систем, де вони використовуються.

По-друге, використання ПЛІС дає можливість значно зменшити час та затрати на проектування схем, розширити можливості розробки модифікацій комп'ютерів, налагодження комп'ютерних пристроїв, що особливо суттєво в стендовому обладнанні, на етапах розробки і виробництва дослідних партій нових виробів, а також емуляції схем.

По-третє, при проектуванні на основі ПЛІС пристроїв для захисту програмного забезпечення виробів від несанкціонованого доступу і копіювання ПЛІС має таку технологічну особливість, як «біт секретності», після програмування якого схема стає недоступною для читання.

Але найбільш широко ПЛІС використовують у мікропроцесорній і обчислювальній техніці. На їх основі розробляють контролери, адресні дешифратори, логіку обладнання мікропроцесора та ін. На основі ПЛІС часто виготовляють мікропрограмні автомати, спеціалізовані пристрої, схеми обробки сигналів і відображення, процесори швидкого перетворення функцій Фур'є і т. д.

Якщо за кордоном ПЛІС уже зайняли належне місце а арсеналі розробника, то в країнах колишнього СРСР ці технології тільки починають по-справжньому розвиватися. Відставання пояснюється рядом причин. По-перше, дуже звужена номенклатура ПЛІС на нашому ринку елементної бази. По-друге, практична відсутність у наших спеціалістів сучасних систем проектування. По-третє, недостатність інформації в технічній літературі про ПЛІС, її застосування і методи програмування.

4.2. Принципи побудови базової програмуємої логічної матриці

Виготовлювані електронною промисловістю ПЛІС мають у собі базову структуру програмованої логічної матриці, яка включає матрицю кон'юнкторів (матриця «І») і матрицю диз'юнкторів (матриця «АБО»). Принцип побудови таких ПЛМ розглянемо на ПЛІС серії K556PT1. Структурна схема даної ПЛІС приведена на рис. 4.2.1.

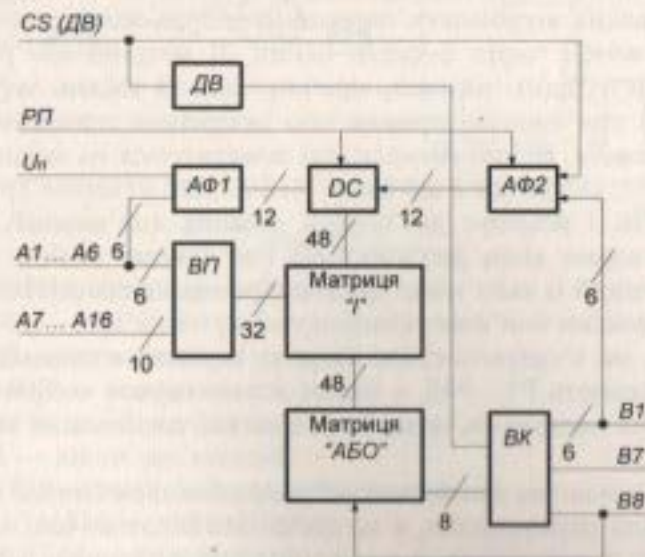


Рис. 4.2.1

Дана ПЛІС включає матрицю кон'юнкторів (матрицю «І»), матрицю диз'юнкторів (матриця «АБО»), блок вхідних підсилювачів (ВП), блок вихідних каскадів (ВК), схему дозволу вибірки кристалу (ДВ), програмований дешифратор, програмовані адресні формувальники (АФ1, АФ2). Вхідні підсилювачі формують прямі й інверсні значення вхідних змінних з усіх шістнадцяти входів (А1...А16).

Програмований дешифратор (ДВ) і програмовані адресні формувальники (АФ1, АФ2) використовують тільки в режимах програмування і контролю ПЛІС. Організація цих режимів достатньо складна і в даному підручнику не розглядається.

Для наочності і більш повного розуміння принципу побудови ПЛІМ розглянемо базову функціональну схему ПЛІС серії К556РТ1, яка включає в себе лише основні вузли схеми матриці «І», «АБО», вхідні і вихідні каскади, рис. 4.2.2.

Вхідні підсилювачі (ВП1...ВП16) формують прямі й інверсні значення вхідних змінних, які надходять у матрицю «І». Для управління вхідними підсилювачами є шістнадцять входів (А1...А16). Вхідні підсилювачі побудовані на основі двох включених послідовно буферних логічних схем «І-НІ».

Основними вузлами мікросхеми К556РТ1 є матриці «І» і «АБО», які реалізують двоохривневі логічні функції. Перший рівень ПЛІМ складається із 48 кон'юнкторів (матриця «І»), які з'єднані за допомогою плавких ніхромових перемичок із будь-яким із шістнадцяти спільних входів через буферні схеми. В матриці «І» реалізують кон'юнкції вхідних змінних, причому кожна вхідна змінна може входити в кон'юнкцію прямим або інверсним значенням, або не входити зовсім. Вхідні сигнали, які pojawiaються на вхідних шинах матриці «І», вводяться в матрицю «АБО», яка утворює другий логічний рівень і реалізує диз'юнкції заданих кон'юнкцій. Матриця «АБО» утворює вісім диз'юнкторів (по одному «АБО» на виході ПЛІС), кожен із яких може бути вибірково з'єднаний із будь-яким із сорока восьми кон'юнкторів.

Шини, які з'єднують ці дві матриці, називають шинами кон'юнкцій і позначають Р1...Р48, а шини, які з'єднують матрицю «АБО» з вихідними каскадами, називають шинами диз'юнкцій та позначають S1...S8.

Програмованим елементом матриці «І» є діод Шотки з плавкою ніхромовою перемичкою, а матриці «АБО» включені за схемою емітерного повторювача, *n-p-n* транзистор із плавкою ніхромовою перемичкою в емітері.

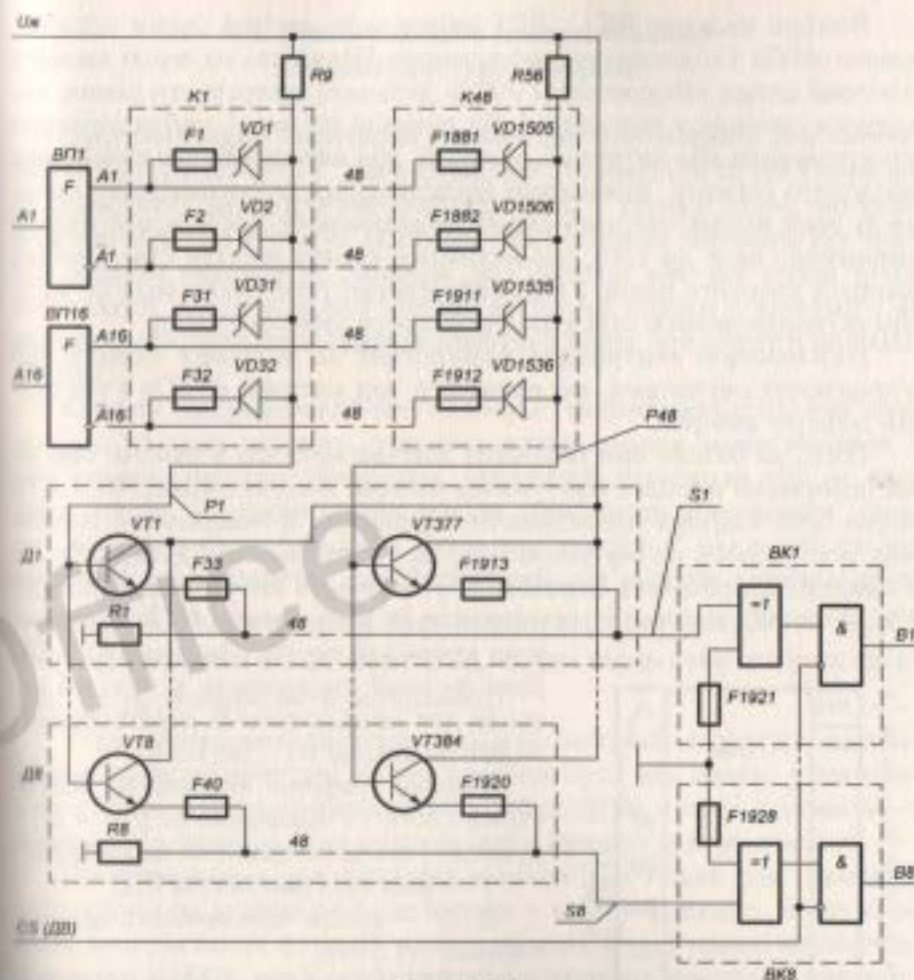


Рис. 4.2.2.

де ВП1...ВП16 — вхідні підсилювачі;
 К1...К48 — кон'юнктори матриці «І»;
 Д1...Д8 — диз'юнктори матриці «АБО»;
 ВК1...ВК8 — вихідні каскади;
 Р1...Р48 — шини кон'юнкцій;
 S1...S8 — шини диз'юнкцій;
 F1...F1928 — плавкі ніхромові перемикачі;
 VD1...VD1536 — діоди Шотки;
 VT1...VT384 — транзистори;
 R1...R56 — резистори.

Вихідні каскади ВК1...ВК8 включають логічні схеми «Виключаюче АБО» і підсилювачі зчитування. Наявність на вході каскаду логічної схеми «Виключаюче АБО» дозволяє інвертувати рівень вихідного сигналу в залежності від сигналу на вході, тобто дозволяє програмувати або активний високий, або активний низький рівень вихідного сигналу. Заземлення (підключення до сигналу «0») одного із двох входів логічної схеми «Виключаюче АБО» через плавку перемикач веде до того, що активним рівнем виходу стає вихідна напруга високого рівня, а виплавлення цієї перемикачки веде до того, що активним рівнем стає вихідна напруга низького рівня.

Підсилювачі зчитування побудовані на логічних схемах, що управляються сигналами, які надходять від матриці «АБО» і від схеми дозволу вибірки.

ПЛІС як базова програмована логічна матриця в режимі обробки інформації працює наступним чином. Вхідні змінні А1...А16 через блок вхідних підсилювачів в прямому й інверсному значеннях поступають на матрицю «І», де за допомогою діодів Шотки і плавких ніхромових перемикачів утворюють потрібні кон'юнкції Р1...Р48, які логічно підсумовуються матрицею «АБО», утворюючи проміжні логічні функції S1...S8.

Дані функції поступають у вихідні каскади для подальшого їх перетворення і видачі на виходи В1...В8 ПЛМ.

Умовне графічне позначення мікросхеми К556РТ1 приведено на рис. 4.2.3, де входи і виходи мікросхеми визначають:

- 1 — вхід програмування РП;
- 2...9 — входи підключення вхідних змінних А1...А8;
- 10...13 — виходи отриманих функцій В8...В5;
- 14 — спільний вихід (вихід подачі «0» В);
- 15...18 — виходи отриманих функцій В4...В1;
- 19 — вхід дозволу роботи (вибору) мікросхеми;
- 20...27 — входи підключення вхідних змінних А16...А9;
- 28 — вхід подачі джерела живлення (+5В).

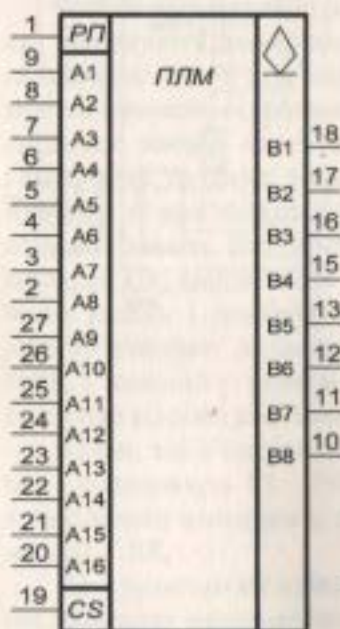


Рис. 4.2.3.

4.3. Рекомендації з програмування базової логічної матриці

Програмування і контроль базової логічної матриці розглянемо на ПЛІС серії К556РТ1. Дана ПЛІС виготовляється та поставляється споживачу не запрограмованою, тобто в такому стані, що кожний кон'юнктор отримує як прямі, так і інверсні значення від кожної вхідної змінної A_i , кожний диз'юнктор має всі сорок вісім кон'юнкцій, а для кожного виходу активним рівнем є високий і на всіх виходах присутня напруга низького рівня при напрузі на вході CS (0В).

Кожний програмований кон'юнктор P_n формує необхідну кон'юнкцію від вхідних змінних, причому кожна змінна може входити в кон'юнкцію прямим значенням, інверсним значенням або не входити зовсім. Ці стани реалізують за допомогою відповідних плавких перемикачів у матриці «І». Якщо кон'юнктор P_n має в собі вхідну змінну A_i , то перемикач, що з'єднує цей кон'юнктор з шиною вхідної змінної $\bar{A}_i$, повинна бути розплавлена, і навпаки. Якщо змінна A_i не має входити в кон'юнктор P_n , то дві перемикачки вхідних змінних A_i і $\bar{A}_i$ повинні бути розплавлені.

Якщо число використаних вхідних змінних A_i менше шістнадцяти, то невикористані змінні повинні бути виключені у всіх використаних кон'юнкторах, тобто відповідні їм плавкі перемикачки в матриці «І» повинні бути розплавлені в процесі програмування.

Програмування диз'юнкторів виконується тільки для тих випадків, коли кон'юнкція не включається у вхідну функцію. Якщо кількість використаних функцій менше восьми, то всі плавкі перемикачки в матриці «АБО», що з'єднує невикористані диз'юнктори і використані або невикористані кон'юнктори переплавляти не потрібно.

4.4. Програмування базової логічної матриці

Програмування базової логічної матриці розглянемо окремо для матриці «І», матриці «АБО» і активного рівня виходу мікросхеми К556РТ1.

Програмування активного рівня виходів В1...В8 відбувається перед програмуванням матриці «І» і матриці «АБО». В початковому стані всі ніхромові перемикачки вихідних каскадів цілі, при цьому рівень вихідного сигналу у вихідному каскаді не інвертується.

ся, і тому рівень активності виходів B1...B8 — високий. Переплавлення одної перемички відбувається при подачі на відповідний вихід напруги $U_{\text{вх.пр}}$. При цьому спрацьовує схема програмування перемички у вихідному каскаді і через перепалювану перемичку проходить руйнуючий її імпульс струму.

При даному програмуванні необхідно:

- виводи 14,1 мікросхеми підключити до 0В, а до виводу 28 подати напругу $0...0,4\text{В}$;
- виводи 10...13, 15...18, крім програмованого, через резистор 10кОм підключити до джерела живлення $5\text{В} \pm 10\%$;
- до виводів 2...9, 19...27 підключити напругу $2,4...4,5\text{В}$;
- на програмований вивід подати напругу $17 \pm 1\text{В}$ і утримувати її $1...5\text{мс}$;
- через $10...15\text{мс}$ після зняття напруги з програмованого виводу, напругу на виводі 28 збільшити до $9,0 \pm 0,5\text{В}$;
- через $10...15\text{мс}$ після збільшення напруги на виводі 28 напругу на виводі 19 зменшити до $0...0,4\text{В}$, на виводах 2, 3, 20...27 установити напругу $0...0,4\text{В}$, на виводах 4...9 — напругу $2,4...4,5\text{В}$, а на програмованому виводі виконати контроль напруги, величина якої при позитивному результаті повинна бути $2,4...4,5\text{В}$.

У випадку негативного результату програмування, тобто при $U_{\text{вх}}^1 = 0...0,4\text{В}$, необхідно ще раз виконати програмування через $t \geq 10\text{мс}$ після закінчення контролю.

Програмування матриці «I» відбувається наступним чином. Для вибору потрібної перемички в мікросхемі є дешифратор DC, рис. 4.2.1, який підключає до джерела програмованого струму відповідну збірку матриці «I». Для управління дешифратором використовують шість адресних формувальників АФ2, адресація яких відбувається із вихідних виводів B1...B6. А це в свою чергу потребує, щоб усі виводи програмованої мікросхеми перебували в закритому стані, для цього на вхід CS необхідно подати напругу $U_{\text{вх.ДВ}}$, що призведе до закриття транзисторів усіх підсилювачів зчитування і на виводи B1...B8 можна подавати адресний код, що відповідає номеру програмованого діодного складання.

Для забезпечення розплавлення тільки потрібної перемички із числа перемичок вибраної дешифратором діодного складання необхідно забезпечити закриття всіх виходів вхідних підсилювачів (як прямих, так і інверсних), крім програмованого. Це забезпечується подачею напруги на входи всіх вхідних підсилювачів, крім

одного. На вході вибраного вхідного підсилювача подають напругу високого рівня $U_{\text{вх}}$, якщо необхідно перепалювати перемичку, з'єднаної з інверсним виходом, або напругу низького рівня $U_{\text{вх}}^0$ — для прямого виходу.

За кожний цикл програмування перепалюється тільки одна перемичка. Імпульс програмованого струму формується при подачі на програмований РП напруги $U_{\text{вх.пр}}$.

При даному програмуванні необхідно:

- вивід 14 мікросхеми підключити до 0В, а на вивід 28 подати напругу $5\text{В} \pm 5\%$;
- на виводі 19 установити напругу $2,4...4,5\text{В}$, а на виводі 1 — $0...0,4\text{В}$;
- виводи 2...9, 2...27, крім програмованого, підключити до джерела $10\text{В} \pm 5\%$;
- на кожний вивід 12, 13, 15...18 (18 — молодший розряд) подати напругу $0...0,4\text{В}$ або $2,4...4,5\text{В}$ у відповідності з кодом адреси кон'юнкції;
- кон'юнкцією включається пряме значення вхідної змінної або напруга $0...0,4\text{В}$, якщо в кон'юнкцію включається інверсне значення вхідної змінної;
- через $10...15\text{мс}$ напруга на виході 1 збільшується до $17 \pm 1\text{В}$ і утримується в процесі всього наступного переходу;
- через $10...15\text{мс}$ напруга на виводі 19 збільшується до $10\text{В} \pm 5\%$ і утримується $1...5\text{мс}$;
- через $10...15\text{мс}$ після зняття напруги на виводі 19 напруга на виводі 1 знижується до $0...0,4\text{В}$;
- через $10...15\text{мс}$ напруга на виводі 19 збільшується до $10\text{В} \pm 5\%$ і на виводі 10 відбувається контроль напруги, величина якої при позитивному результаті програмування повинна бути $2,4...4,5\text{В}$.

У випадку негативного результату програмування ($0...0,4\text{В}$) необхідно виконати повторне програмування шляхом його одно-, двохкратного повторення через $t \geq 10\text{мс}$ після закінчення контролю.

Якщо вхідна змінна і при цьому не включається в кон'юнкцію, то її необхідно виключити із кон'юнкції шляхом подачі на програмований вхід напруги $2,4...4,5\text{В}$, а потім $0...0,4\text{В}$ і вхідних значень згідно з описаним вище.

Програмування матриці «АБО» відбувається наступним чином. В початковому стані всі ніхромові перемички матриці «АБО» цілі. Для формування потрібних функцій необхідно в кожний із них

включити ті кон'юнкції, які не повинні входити у відповідну функцію, тобто розплавити деякі перемички матриці.

Для програмування матриці «АБО» використовують той же дешифратор DC, рис. 4.2.1, що і при програмуванні матриці «І», але управління ним відбувається через другі групи програмованих адресних формувальників АФ1 зі сторони входних виводів А1...А6. Підключення АФ1 до джерела живлення і установка входних підсилювачів в необхідний стан відбувається при подачі на мікросхему збільшеної напруги. На виводи А1...А6 подається код, відповідний номеру логічного добутку, який необхідно виключити із даної функції, а на вхід «CS» (DB) — напругу 2,4...4,5В, яка встановлює виходи всіх підсилювачів зчитування в закритий стан. На вихід відповідної функції із якої виключається вибрана кон'юнкція, подається напруга $U_{\text{вих.ф.}}$. Імпульс програмованого струму, який протікає за вибраною перемичкою, формується при подачі на програмований вхід РП напруги $U_{\text{вих.пр.}}$, а на вхід CS (DB) — $U_{\text{вих.DB}}$. За кожного цикл програм програмується тільки одна перемичка.

При даному програмуванні необхідно:

- вивід 14 підключити до 0 В, вивід 28 — до $9,0\text{В} \pm 0,5\text{ В}$, вивід 19 — до 2,4...4,5 В, а виводи 1...3, 20...27 — до 0...0,4 В;
- на виводи 4...9 подати напругу 0...0,4 В і 2,4...4,5 В у відповідності з кодом адреси кон'юнкції, яку не включають у вхідну функцію;
- виводи 10...13, 15...18, крім програмованого, підключити до джерела постійної напруги $4,5\text{ В} \pm 10\%$;
- на програмованому виводі виставляється напруга $4,5\text{ В} \pm 5\%$;
- через 10...15 мкс після подачі напруги на програмований вивід напруга на виводі 1 збільшується до $10\text{ В} \pm 5\%$ і утримується до наступного переходу;
- через 10...15 мкс після збільшення напруги на виводі 1 напруга на виводі 19 збільшується до $10\text{ В} \pm 5\%$ і утримується 1...5 мс, після чого зменшується до 2,4...4,5 В;
- через 10...15 мкс після зняття напруги на виводі 19 напруга на виводі 1 знижується до 0...0,4 В;
- через 10...15 мкс, після зниження напруги на виводі 1 напруга зовнішнього джерела $10\text{ В} \pm 5\%$ від програмованого виводу відключається, а на вивід 19 виставляється напруга 0...0,4 В, після чого на програмованому виводі відбувається контроль напруги, величина якої при задовільному результаті програмування повинна бути

2,4...4,5 В для виводу з активним низьким рівнем або 0...0,4 В для виводу з активним високим рівнем.

У випадку негативного результату програмування, тобто при 0...0,4 В для виводу з активним рівнем, необхідно повторити його через $t \geq 10\text{ мс}$ після контролю згідно з наведеною вище програмою.

4.5. Схемотехніка побудови комбінаційних схем на програмованих логічних матрицях

На ПЛМ зручно будувати як одновихідні, так і особливо багатовихідні комп'ютерні комбінаційні схеми.

Схемотехніка побудови комбінаційних схем на програмованих логічних матрицях має такий алгоритм. На першому кроці алгоритму визначають кількість входних і вихідних змінних, а також кон'юнкцій, які необхідно реалізувати на ПЛМ. На другому кроці, використовуючи дані логічних функцій першого кроку алгоритму, визначають дану ПЛМ (їх сукупність) для реалізації заданих функцій. На третьому кроці, згідно з алгоритмом роботи системи, визначають, яку кількість логічних функцій необхідно отримати з високим рівнем активності, а яку — з низьким. На четвертому кроці алгоритму заданим логічним функціям присвоюють номери їх кон'юнкторів. На п'ятому кроці, використовуючи рекомендації із програмування — §4.3 і дані — §4.4, програмують утворені на четвертому кроці кон'юнктори логічних функцій, а також і самі функції. Невикористані входні входи ПЛМ повинні бути виключені у всіх використаних кон'юнкторах шляхом розплавлення їх ніхромових перемичок у матриці «І» в процесі програмування. На шостому кроці алгоритму контролюють правильність програмування ПЛМ, і якщо воно відбулося правильно, то запрограмовану мікросхему використовують для реалізації заданих логічних функцій, а якщо ні, то необхідно повернутись до п'ятого кроку алгоритму.

Приклад 4.5.1. Побудувати (реалізувати) на ПЛМ K556PT1 таку систему логічних функцій

$$f_1 = x_2 \cdot \bar{x}_4 \cdot x_5 \vee \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 \cdot x_4 \vee x_1 \cdot \bar{x}_2 \cdot x_3 \cdot \bar{x}_4 \vee x_1 \cdot \bar{x}_3 \cdot \bar{x}_5;$$

$$f_2 = x_1 \cdot \bar{x}_2 \cdot x_3 \vee x_2 \cdot \bar{x}_3 \cdot \bar{x}_4 \cdot x_5 \vee \bar{x}_1 \cdot x_2 \cdot x_4 \cdot \bar{x}_5;$$

$$f_3 = \bar{x}_1 \cdot \bar{x}_4 \cdot x_5 \vee x_4 \cdot \bar{x}_5 \vee x_2 \cdot \bar{x}_4 \cdot \bar{x}_5 \vee x_3 \cdot x_4 \cdot \bar{x}_5;$$

$$f_4 = x_5 \cdot \bar{x}_4 \cdot \bar{x}_5 \vee x_1 \cdot \bar{x}_2 \cdot x_3 \vee x_1 \cdot \bar{x}_4 \cdot x_5 \vee x_1 \cdot \bar{x}_2 \cdot \bar{x}_4 \cdot x_5;$$

з високим рівнем активності для перших двох функцій і низьким для двох останніх.

Розв'язання. Для реалізації цієї системи логічних функцій на ПЛМ згідно з першим кроком визначаємо необхідну кількість входів, виходів і кон'юнкцій для програмування їх у мікросхемі К556РТ1. Кількість входів дорівнює 5, виходів — 4, а кон'юнкцій — 15. Так як задана ПЛМ має 16 входів, 8 виходів і може реалізувати 48 кон'юнкцій, §4.2, то робимо висновок, що на ній можливо запрограмувати задану систему логічних рівнянь, що відповідає другому кроку.

Згідно із заданими функціями $f_1, \dots, f_4$ присвоюємо номери їх кон'юнкторів:

$$k_1 = x_2 \cdot \bar{x}_4 \cdot x_5;$$

$$k_2 = \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 \cdot x_4;$$

$$k_3 = \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 \cdot x_4;$$

$$k_4 = x_1 \cdot \bar{x}_3 \cdot \bar{x}_5;$$

$$k_5 = x_1 \cdot \bar{x}_2 \cdot x_3;$$

$$k_6 = x_2 \cdot \bar{x}_3 \cdot \bar{x}_4 \cdot x_5;$$

$$k_7 = \bar{x}_1 \cdot x_2 \cdot x_4 \cdot \bar{x}_5;$$

$$k_8 = \bar{x}_1 \cdot \bar{x}_4 \cdot x_5;$$

$$k_9 = x_4 \cdot \bar{x}_5;$$

$$k_{10} = x_2 \cdot \bar{x}_4 \cdot \bar{x}_5;$$

$$k_{11} = x_3 \cdot x_4 \cdot \bar{x}_5;$$

$$k_{12} = x_3 \cdot \bar{x}_4 \cdot \bar{x}_5;$$

$$k_{13} = x_1 \cdot \bar{x}_2 \cdot x_3;$$

$$k_{14} = x_1 \cdot \bar{x}_4 \cdot x_5;$$

$$k_{15} = x_1 \cdot \bar{x}_2 \cdot \bar{x}_4 \cdot x_5.$$

Використовуючи рекомендації із програмування, наведені в §4.3 і дані §4.4, програмуємо задані функції і їх дані заносимо в табл. 4.5.1.

Таблиця 4.5.1

Номер кон'юнктора	Кон'юнктори					Рівень активності			
	Вхідна змінна					1	1	0	0
	x_1	x_2	x_3	x_4	x_5	Вихідна функція			
	Номер програмованого входу					f_1	f_2	f_3	f_4
	A1	A2	A3	A4	A5	B1	B2	B3	B4
I	2	3	4	5	6	7	8	9	10
$k_1 = x_2 \cdot \bar{x}_4 \cdot x_5$	*	1	*	0	1	A	*	*	*
$k_2 = \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 \cdot x_4$	0	1	0	1	*	A	*	*	*
$k_3 = \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 \cdot x_4$	1	0	1	0	*	A	*	*	*

Закінчення табл. 4.5.1

Номер кон'юнктора	Кон'юнктори					Рівень активності			
	Вхідна змінна					1	1	0	0
	x_1	x_2	x_3	x_4	x_5	Вихідна функція			
	Номер програмованого входу					f_1	f_2	f_3	f_4
	A1	A2	A3	A4	A5	B1	B2	B3	B4
I	2	3	4	5	6	7	8	9	10
$k_4 = x_1 \cdot \bar{x}_3 \cdot \bar{x}_5$	1	*	0	*	0	A	*	*	*
$k_5 = x_1 \cdot \bar{x}_2 \cdot x_3$	1	0	1	*	*	*	A	*	*
$k_6 = x_2 \cdot \bar{x}_3 \cdot \bar{x}_4 \cdot x_5$	*	1	0	0	1	*	A	*	*
$k_7 = \bar{x}_1 \cdot x_2 \cdot x_4 \cdot \bar{x}_5$	0	1	*	1	0	*	A	*	*
$k_8 = x_1 \cdot \bar{x}_3 \cdot \bar{x}_5$	0	*	*	0	1	*	*	A	*
$k_9 = x_4 \cdot \bar{x}_5$	*	*	*	1	0	*	*	A	*
$k_{10} = x_2 \cdot \bar{x}_4 \cdot \bar{x}_5$	*	1	*	0	0	*	*	A	*
$k_{11} = x_3 \cdot x_4 \cdot \bar{x}_5$	*	*	1	1	0	*	*	A	*
$k_{12} = x_3 \cdot \bar{x}_4 \cdot \bar{x}_5$	*	*	1	0	0	*	*	*	A
$k_{13} = x_1 \cdot \bar{x}_2 \cdot x_3$	1	0	1	*	*	*	*	*	A
$k_{14} = x_1 \cdot \bar{x}_4 \cdot x_5$	1	*	*	0	0	*	*	*	A
$k_{15} = x_1 \cdot \bar{x}_2 \cdot \bar{x}_4 \cdot x_5$	1	0	*	0	1	*	*	*	A

При програмуванні кон'юнкторів змінну x_i , яка входить в кон'юнкцію прямим значенням, програмують «1», якщо інверсним то — «0», а якщо не входить зовсім, то — «*», що означає перепалвлення перемички.

При програмуванні диз'юнкторів належність k_j до даної функції помічають у таблиці знаком «A», а його відсутність позначають знаком «*», що означає перепалвання перемички.

Високий рівень активності виходу програмується «1», а низький — «0».

На всіх невикористаних входах A6...A16 і виходах B5...B8 ПЛМ із номерами кон'юнкторів K1...K15 перемички перепалюються.

Схема, яка реалізує систему рівнянь умови прикладу 4.5.1, приведена на рис. 4.5.1.

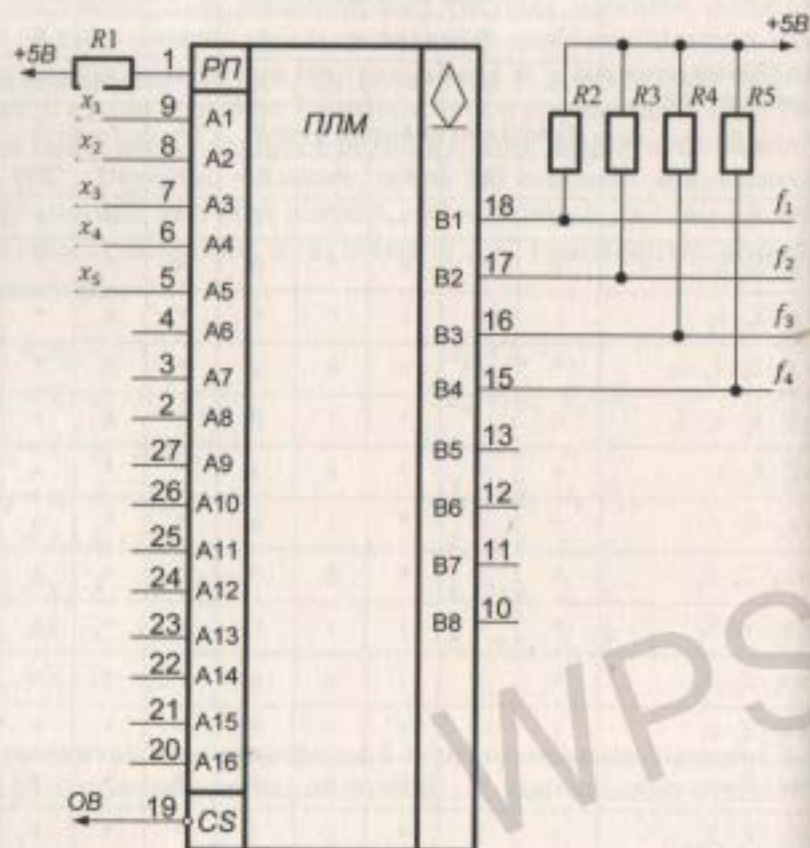


Рис. 4.5.1

де $R1 \dots R5$ — резистор 1 кОм.

Як впливає із табл. 4.5.1 і рис. 4.5.1, реалізація будь-яких логічних функцій дуже зручна на ПЛМ.



Контрольні запитання

1. Яке призначення має програмована логічна матриця?
2. Які ділянки застосування програмованої логічної матриці?
3. Для яких цілей застосовують програмовані логічні матриці?

4. З яких блоків складається програмована логічна матриця?
5. Яка максимальна кількість кон'юнкторів може бути запрограмована в ПЛІС K556PT1?
6. Яка максимальна кількість логічних функцій може бути запрограмована в ПЛІС K556PT1?
7. Який процес відбувається при програмуванні в ПЛІС K556PT1?
8. Які дві матриці є обов'язковими у базовій програмованій матриці, назвіть їх?
9. В якій формі можна реалізовувати логічні функції у базовій програмованій логічній матриці і чому?
10. Назвіть переваги побудови комбінаційних схем на програмованих логічних матрицях перед побудовою комбінаційних логічних схем на елементах елементарних булевих функцій.



Коментарі. В даному розділі використаний матеріал взято з [17, 19].



Розділ 5

СХЕМОТЕХНІКА ПОБУДОВИ ТИПОВИХ СХЕМ ІЗ ПАМ'ЯТТЮ

5.1. Схемотехніка побудови RS-тригерів

Означення 5.1.1. RS-тригером називають запам'ятовувальний пристрій із двома стійкими станами і з різними інформаційними входами для установки його в стан «0» (R — вхід) і стан «1» (S — вхід). Назву RS-тригера утворено від перших букв слів англійської мови *Reset* (виключити) і *Set* (включити).

Алгоритм побудови RS-тригера виконують за п'ять кроків. На першому кроці, досліджуючи логіку роботи RS-тригера, будують таблицю його переходів. На другому кроці, використовуючи таблицю переходів, будують карти Карно, а на третьому, за допомогою карт Карно, отримують логічні рівняння роботи RS-тригера, які на четвертому кроці алгоритму за допомогою законів алгебри логіки перетворюють до вигляду, на якому корисно його реалізувати, застосувавши один із видів основних логічних елементів, наведених у §2.6. На п'ятому кроці алгоритму, за отриманим на четвертому кроці логічним рівнянням, будують RS-тригер на вибраній елементній базі.

Згідно з першим кроком алгоритму, досліджуючи логіку роботи RS-тригера, будують таблицю його переходів, табл. 5.1.1.

Таблиця 5.1.1

R_i	S_i	Q_i	Q_{i+1}
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	K_1
1	1	1	K_2

В таблиці переходів прийняті наступні позначення: R_i, S_i, Q_i — значення логічних змінних в момент часу t на входах R, S і виході Q тригера; Q_{i+1} — стан тригера після переключення; K_1, K_2 — невизначені коефіцієнти на тих наборах, де вхідні сигнали R_i і S_i одночасно приймають значення одиниці (заборонена комбінація сигналів).

На другому кроці, користуючись таблицею переходів RS-тригера будують карти Карно, які приведені на рис. 5.1.1а

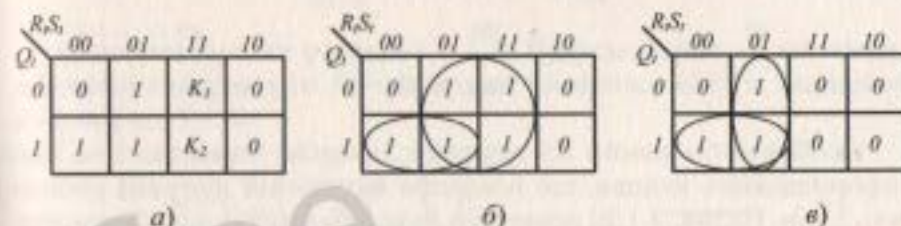


Рис. 5.1.1

Припустимо, що комбінації вхідних сигналів $R_i S_i = 11$ не має, тоді отримаємо карти Карно для $K_1 = K_2 = 1$, рис. 5.1.1б, і $K_1 = K_2 = 0$, рис. 5.1.1в.

На третьому кроці алгоритму побудови RS-тригера із карт Карно, рис. 5.1.1а і рис. 5.1.1б, отримаємо логічні рівняння асинхронного RS-тригера

$$Q_{i+1} = S_i \vee \bar{R}_i \cdot Q_i \quad \text{при } K_1 = K_2 = 1, \quad (5.1.1)$$

$$Q_{i+1} = \bar{R}_i (S_i \vee Q_i) \quad \text{при } K_1 = K_2 = 0. \quad (5.1.2)$$

Логічні рівняння 5.1.1 і 5.1.2 визначають новий стан тригера Q_{i+1} в залежності від старого стану Q_i і вхідних сигналів R_i і S_i .

На четвертому кроці алгоритму побудови RS-тригера перетворюємо логічне рівняння 5.1.1 до вигляду, на якому корисного його реалізувати на елементах «І-НІ»:

$$Q_{i+1} = S_i \vee \bar{R}_i \cdot Q_i = \overline{\overline{S_i} \cdot \overline{\bar{R}_i \cdot Q_i}} = \overline{\overline{S_i} \cdot \overline{\bar{R}_i} \cdot \overline{Q_i}}. \quad (5.1.3)$$

Використовуючи логічні зв'язки рівняння 5.1.3, на п'ятому кроці алгоритму будують схему синхронного RS-тригера на елементах «І-НІ», яка матиме вигляд, наведений на рис. 5.1.2а.

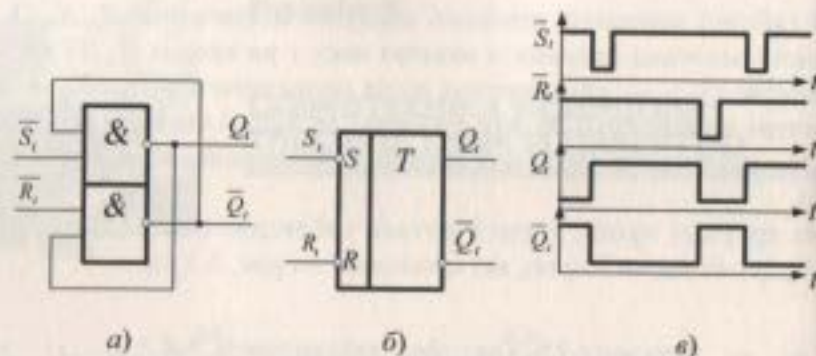


Рис. 5.1.2

Особенностью данного RS -триггера є інверсне управління на його інформаційних входах, що показано на часовій діаграмі роботи, рис. 5.1.2в. На рис. 5.1.2б приведена функціональна схема RS -триггера.

Для побудови RS -триггера на елементах «АБО-НІ» необхідно, згідно з кроком четвертим алгоритму, перетворити логічні рівняння 5.1.2 до такого вигляду

$$Q_{i+1} = \bar{R}_i \cdot (S_i \vee Q_i) = \bar{R}_i \cdot (S_i \vee Q_i) = R_i \vee (\bar{S}_i \cdot \bar{Q}_i). \quad (5.1.4)$$

Тоді, використовуючи логічні зв'язки рівняння 5.1.4, на п'ятому кроці алгоритму будуємо схему асинхронного RS -триггера на елементах «АБО-НІ», наведену на рис. 5.1.3а.

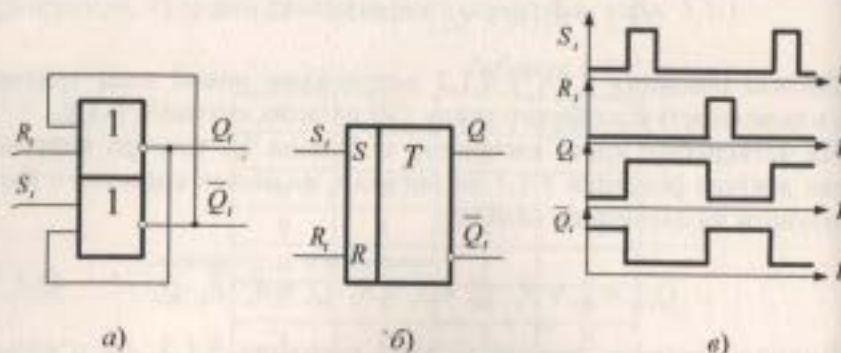


Рис. 5.1.3

Часова діаграма роботи даного триггера приведена на рис. 5.1.3в, а зображення функціональної схеми — на рис. 5.1.3б.

Для побудови синхронного RS -триггера необхідно в рівнянні 5.1.3 ввести змінну сигналу синхронізації C_i .

У логічному рівнянні 5.1.3 змінні S_i і R_i скомпонуємо із сигналом синхроімпульсу C_i наступним чином: $C_i \cdot S_i$ і $C_i \cdot R_i$, в результаті чого отримаємо логічне рівняння

$$Q_{i+1} = \bar{C}_i \cdot \bar{S}_i \cdot \bar{C}_i \cdot \bar{R}_i \cdot Q_i. \quad (5.1.5)$$

Використовуючи рівняння 5.1.5, будуємо схему з логічними зв'язками синхронного RS -триггера на елементах «І-НІ», яка приведена на рис. 5.1.4а.

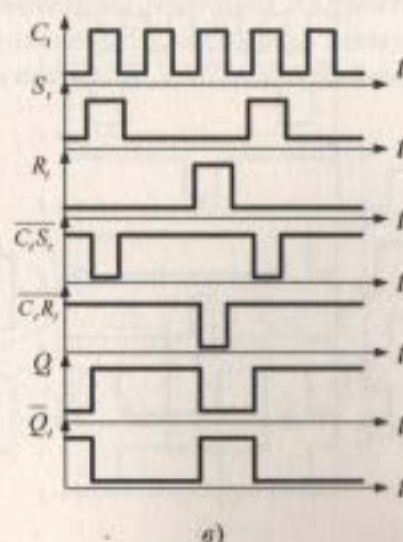
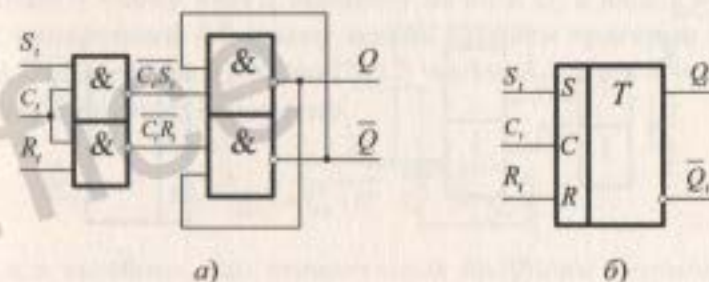


Рис. 5.1.4

Часова діаграма роботи D -тригера приведена на рис. 5.2.1а, а функціональна схема — на рис. 5.2.1б. Із часової діаграми випливає, що D -тригер слідує за змінною сигналу на D -вході в час дії сигналу синхронізації C , і зберігає ту інформацію, яка була в момент його кінця.

З логіки функціонування D -тригера, рівняння 5.2.1 також впливає друга схема D -тригера, яка побудована з використанням функціональної схеми синхронного RS -тригера й інвертора, рис. 5.2.2.

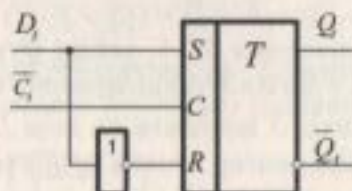


Рис. 5.2.2

Часова діаграма роботи цієї схеми аналогічна часовій діаграмі, приведений на рис. 5.2.1а.

5.3. Схемотехніка побудови T -тригера

Означення 5.3.1. T -тригером називають запам'ятовуючий пристрій із двома стійкими станами, які міняються на протилежні після кожного надходження сигналу на його інформаційний вхід T .

Схемотехніку побудови T -тригера виконують за чотири кроки алгоритму. На першому кроці алгоритму, досліджуючи його роботу, будують таблицю переходів. На другому, із таблиці переходів, отримують логічні рівняння роботи T -тригера, які на третьому кроці алгоритму за допомогою законів алгебри логіки перетворюють на вигляд, в якому корисно його реалізувати, застосовуючи один із видів основних логічних елементів, наведених у §2.6. На четвертому кроці алгоритму, за отриманим логічним рівнянням, будують T -тригер на вибраній елементній базі.

Згідно з першим кроком алгоритму, досліджуючи логіку функціонування T -тригера будуємо таблицю його переходів, табл. 5.3.1.

Таблиця 5.3.1

T_i	Q_i	Q_{i+1}
0	0	0
0	1	1
1	0	1
1	1	0

Із таблиці переходів випливає логічне рівняння роботи T -тригера, яке має вигляд:

$$Q_{i+1} = \overline{T}_i \cdot Q_i \vee T_i \cdot \overline{Q}_i. \quad (5.3.1)$$

На третьому кроці побудови T -тригера, використовуючи закони алгебри логіки (подвійного заперечення і закон де-Моргана), перетворюємо рівняння 5.3.1 на вигляд:

$$Q_{i+1} = \overline{T}_i \cdot Q_i \vee T_i \cdot \overline{Q}_i = \overline{\overline{\overline{T}_i \cdot Q_i} \cdot \overline{T_i \cdot \overline{Q}_i}}. \quad (5.3.2)$$

В рівнянні 5.3.2 для виключення інверсії сигналу T , використовуємо наступну логічну тотожність $\overline{T}_i \cdot Q_i = (\overline{T_i \cdot \overline{Q}_i}) \cdot Q_i$. Підставивши її в рівняння 5.3.2, отримаємо

$$Q_{i+1} = \overline{\overline{\overline{T_i \cdot \overline{Q}_i}} \cdot \overline{T_i \cdot \overline{Q}_i}}. \quad (5.3.3)$$

На четвертому кроці будемо схему T -тригера на елементах «І-НІ» з використанням рівняння 5.3.3, рис. 5.3.1а.

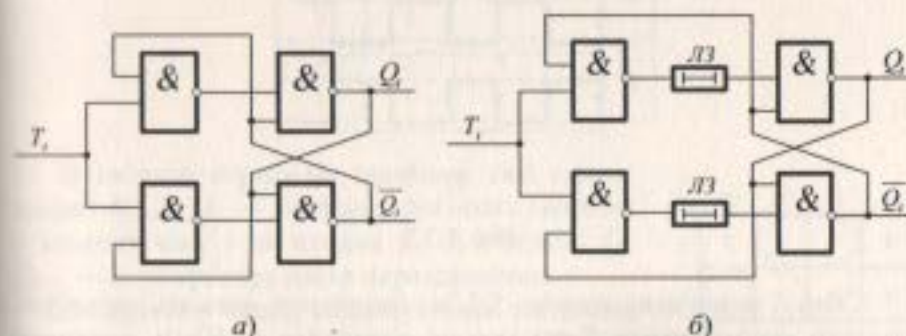
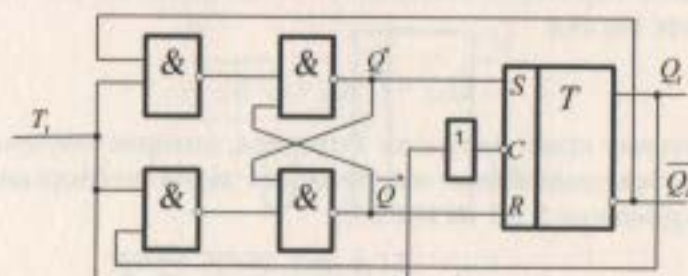
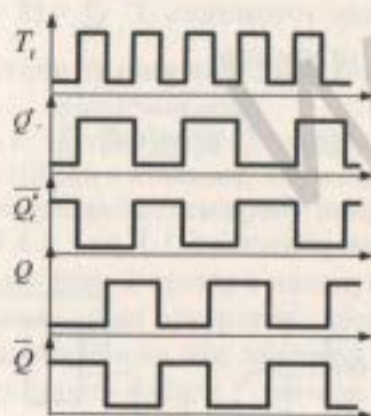


Рис. 5.3.1

Із аналізу роботи схеми T -тригера (рис. 5.3.1а) слідує, що сигнали оберненого зв'язку в тригері міняються в час дії сигналу T_i . Тобто робота елемента пам'яті T -тригера є не стійкою (може виникнути коливальний режим). Уникнути цього явища можна двома шляхами. Перший шлях полягає у затримці сигналів, які надходять на елемент пам'яті T -тригера, на час дії сигналу T_i , використовуючи для цього лінію затримки (ЛЗ), а при другому необхідно застосовувати додатковий асинхронний RS -тригер. Схема T -тригера, яка реалізує перший шлях приведена на рис. 5.3.1б, а другий — на рис. 5.3.2а.



а)



б)

Рис. 5.3.2

Схему, приведену на рис. 5.3.2а, називають схемою двохступінчатого асинхронного T -тригера на елементах «І-НІ» з логічними зв'язками, відповідно до рівняння 5.3.3. На рис. 5.3.2б приведена

часова діаграма роботи даної схеми, яка є стійкою в роботі і виключає різні «гонки» в схемі за рахунок затримки сигналів на її елементах.

5.4. Схемотехніка побудови JK -тригера

Означення 5.4.1. JK -тригером називають запам'ятовуючий пристрій із двома стійкими станами й інформаційними входами J (аналог S), K (аналог R), які забезпечують незалежну установку станів «1» і «0», а при співпадінні сигналів $JK = \{1, 0\}$ він переключастись в протилежні стани, тобто реалізує додавання сигналів за модулем два.

Схемотехніку побудови JK -тригера виконують за чотири кроки алгоритму. На першому кроці алгоритму, досліджуючи логіку JK -тригера, будують таблицю його переходів. На другому кроці за таблицею переходів отримують карти Карно, за яким на третьому отримують логічне рівняння роботи JK -тригера, а на четвертому за отриманим логічним рівнянням будують JK -тригер на вибраній елементній базі.

Згідно з першим кроком алгоритму, досліджуючи логіку роботи JK -тригера, будують таблицю його переходів, табл. 5.4.1.

Таблиця 5.4.1

K_i	J_i	Q_i	Q_{i+1}
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

В таблиці переходів прийняті такі позначення: K_i, J_i, Q_i — значення логічних змінних у момент часу t на входах K_i, J_i і виході Q_i ; Q_{i+1} — стан тригера після переключення.

На другому кроці, користуючись таблицею переходів JK -тригера, будують карту Карно, рис. 5.4.1.

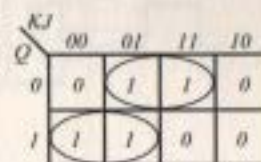


Рис. 5.4.1

Із карти Карно отримаємо логічне рівняння роботи JK -тригера

$$Q_{i+1} = \bar{K}_i \cdot Q_i \vee J_i \cdot \bar{Q}_i; \quad (5.4.1)$$

яке за законами алгебри логіки (закон подвійного заперечення і закон де-Моргана) перетворимо в такий вигляд:

$$Q_{i+1} = \bar{K}_i \cdot Q_i \vee J_i \cdot \bar{Q}_i = \overline{\overline{\bar{K}_i \cdot Q_i \vee J_i \cdot \bar{Q}_i}} \quad (5.4.2)$$

Для виключення заперечення сигналу K_i в рівнянні 5.4.2 використовуємо логічну тотожність $\bar{K}_i \cdot Q_i = (\overline{K_i \cdot Q_i}) \cdot Q_i$. Підставивши її в рівняння 5.4.2, отримаємо

$$Q_{i+1} = \overline{(K_i \cdot Q_i) \cdot Q_i \cdot J_i \cdot \bar{Q}_i} \quad (5.4.3)$$

Для побудови синхронного JK -тригера на елементах «І-НІ» необхідно замінити в рівнянні 5.4.3 змінні K_i і J_i на $C_i \cdot K_i$ і $C_i \cdot J_i$ відповідно. В результаті цього отримаємо

$$Q_{i+1} = \overline{(C_i \cdot K_i \cdot Q_i) \cdot Q_i \cdot C_i \cdot J_i \cdot \bar{Q}_i} \quad (5.4.4)$$

На четвертому кроці, використовуючи логічні зв'язки рівняння 5.4.4, будемо одноступінчатий асинхронний JK -тригер, який приведений на рис. 5.4.2а

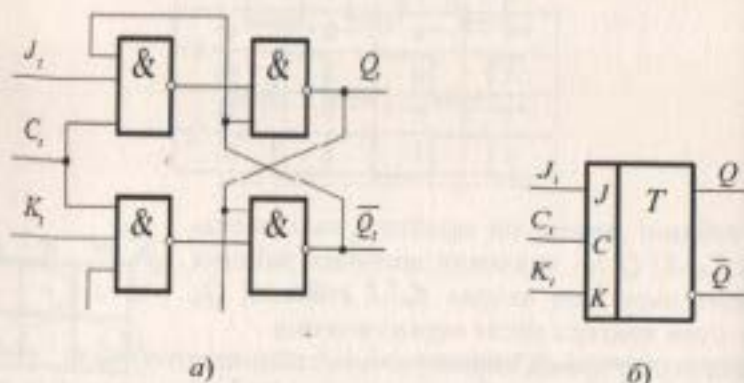


Рис. 5.4.2

На рис. 5.4.2б приведена функціональна схема одноступінчатого JK -тригера. Дана схема працює надійно в режимі роз'єднаних входів J і K , тобто в режимі RS -тригера, а при їх об'єднанні, тобто в режимі T -тригера необхідно використати один із двох шляхів, ліквідації «гонок» сигналів, які були розглянуті в §5.3. На практиці, як правило, застосовують двохступінчаті синхронні JK -тригери. Схема такого тригера із застосуванням рівняння 5.4.4 в кожній ступені, приведена на рис. 5.4.3а, його функціональна схема на рис. 5.4.3б, а часова діаграма роботи в режимі RS -тригера — на рис. 5.4.4.

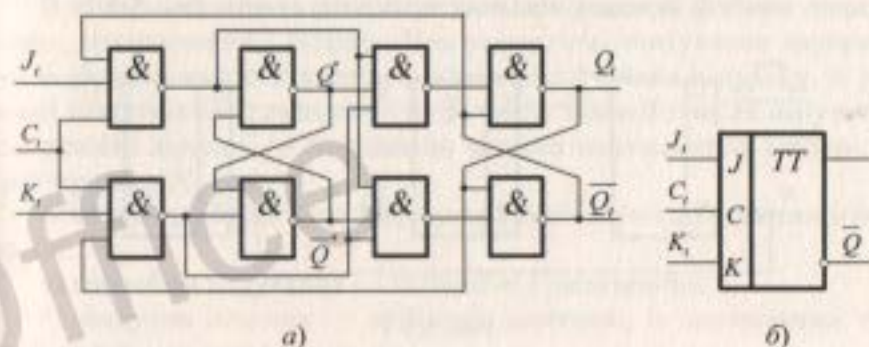


Рис. 5.4.3

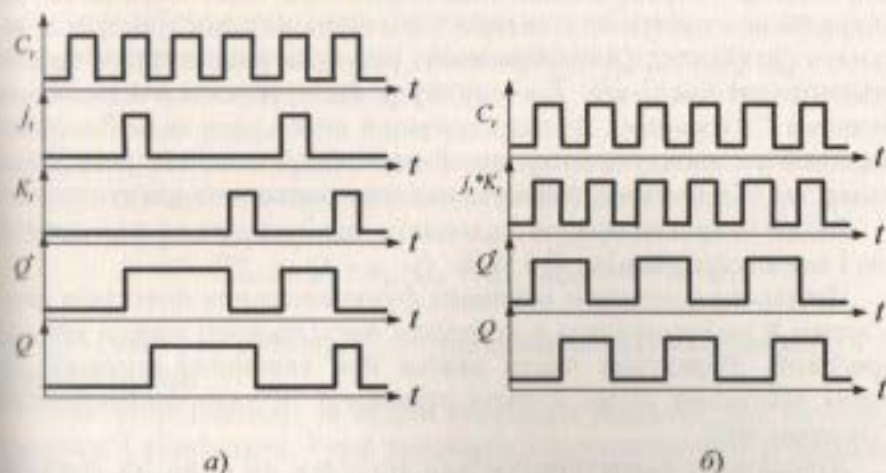


Рис. 5.4.4

На рис. 5.4.46 приведена часова діаграма роботи двохступінчатого JK -тригера при умові, коли входи J_i і K_i — об'єднані, тобто JK -тригер уключений у лічильному режимі.

5.5. Схемотехніка побудови лічильників

Означення 5.5.1. Лічильником називають функціональний пристрій комп'ютера, призначений для підрахунку вхідних імпульсів.

Лічильники являють собою зв'язаний ланцюг T -тригерів, утворюють пам'ять із заданою кількістю сталих станів, рис. 5.5.1

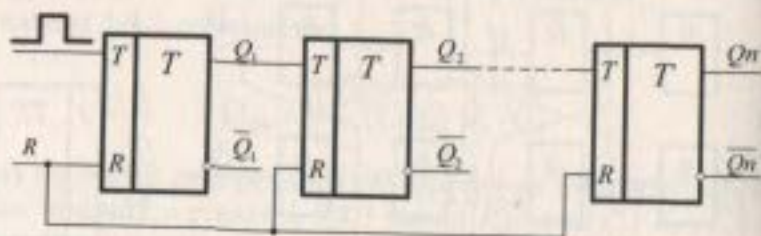


Рис. 5.5.1

Розрядність лічильника n дорівнює кількості T -тригерів. Кожний вхідний імпульс змінює стан лічильника, який зберігається до надходження наступного сигналу. Значення виходів тригерів лічильника $Q_n, Q_{n-1}, \dots, Q_1$ відображають результат підрахунку в прийнятій системі числення. Логічну функцію лічильника позначають буквами CT (counter). До мікрооперацій лічильника включають попередню установку в початковий стан, інкремент або декремент слова, що зберігається, видачу слова паралельним кодом та ін.

Вхідні імпульси можуть надходити на лічильник як періодично, так і задовільно розміщені в часі.

Лічильник є одним із основних функціональних пристроїв комп'ютера й застосовується в утворенні послідовності адреси команди програми, підрахунку числа циклів при виконанні операцій ділення, множення, зсуву, а також отримання сигналів мікрооперацій і синхронізації.

Лічильники характеризуються модулем та ємністю лічення. Модуль лічення $K_{\text{ем}}$ визначає кількість станів лічильника. Модуль двійкового n -розрядного лічильника визначається цілим степенем

двійки $M = 2^n$. В лічильниках інших типів справедлива нерівність $K_{\text{ем}} \leq M$. Після підрахунку кількості імпульсів $N_{\text{ем}} = K_{\text{ем}}$ лічильник повертається в початковий стан. Таким чином, модуль лічення, який часто називають коефіцієнтом перерахунку, визначає цикл роботи лічильника, після якого його стан повторюється. Тому кількість вхідних імпульсів і стан лічильника однозначно визначені тільки для першого циклу.

Ємність лічення $N_{\text{макс}}$ визначається максимальною кількістю вхідних імпульсів, які може зафіксувати лічильник при одному циклі роботи.

В лічильниках використовуються три режими роботи: управління, накопичення і ділення. При управлінні зчитування інформації відбувається після кожного вхідного зліченого імпульсу. В режимі накопичення головним є підрахунок заданого числа імпульсів, а в режимі ділення — зменшення частоти надходження імпульсів, наприклад, у $K_{\text{ем}}$ раз.

Лічильники за логікою побудови класифікують за такими ознаками:

- а) способом кодування — позиційні і непозиційні;
- б) модулем лічення — двійкові, десяткові, із задовільним постійним або змінним модулем;
- в) напрямком лічення — прості (підсумовуючі, віднімаючі) та реверсивні;
- г) способом організації мікрозрядних зв'язків — із послідовним, наскрізним, паралельним і комбінованим переносами;
- д) типом використаних тригерів — T , JK , D в лічильному режимі.

В лічильниках з позиційним кодуванням числове значення поточного стану лічильника визначається за формулою

$$N = \sum_{i=1}^n a_i \cdot Q_i = a_n \cdot Q_n + a_{n-1} \cdot Q_{n-1} + \dots + a_1 \cdot Q_1,$$

де a_i — вага i -го розряду; Q_i — значення виходу i -го розряду; n — число розрядів.

Прості лічильники за видом переходів розділяються на підсумовуючі і віднімаючі. Граф переходів підсумовуючого лічильника приведений на рис. 5.5.2а, віднімаючого — на рис. 5.5.2б, а реверсивного — на рис. 5.5.2в.

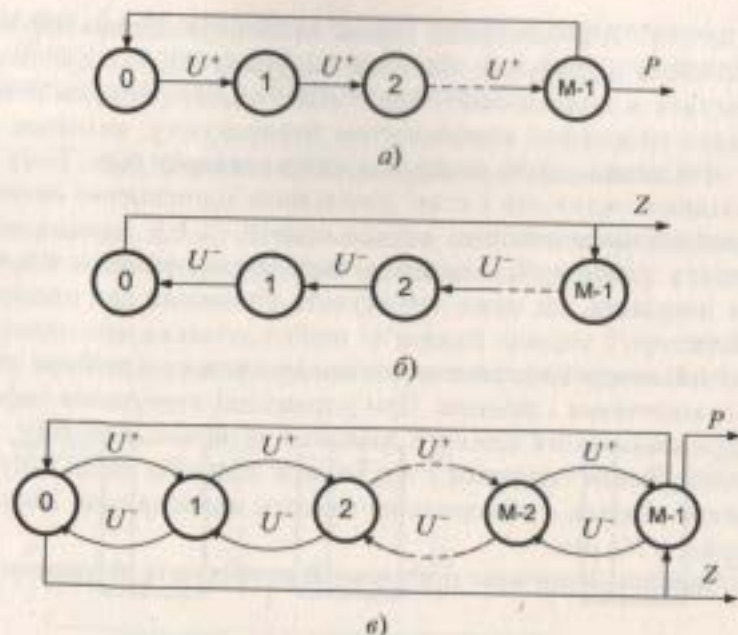


Рис. 5.5.2

Реверсивні лічильники мають переходи в прямому і оберненому напрямках, що дозволяє рахувати додаємі і віднімаємі імпульси, що додаються і віднімаються рис. 5.5.2в. У процесі лічення повинна виконуватись умова $\sum U^+ + N_n \geq \sum U^-$, де N_n — попередньо записане число. За поточним станом виходів лічильника визначають результат реверсивного лічильника

$$\Delta N = \sum U^+ + N_n - \sum U^-.$$

Використовуючи рис. 5.5.2, розглянемо побудову асинхронних двійкових підсумовуючих, віднімаючих і реверсивних лічильників із використанням T -тригерів.

Асинхронні підсумовуючі лічильники на двохступінчатих T -тригерах будують таким чином, щоб вхідні імпульси U^+ надходили тільки на лічильний вхід першого розряду. Сигнали переносу в лічильнику передаються асинхронно з прямих виходів молодших розрядів на T -входи сусідніх старших. Схема такого лічильника на три розряди приведена на рис. 5.5.3а, а часова діаграма його роботи — на рис. 5.5.3б.

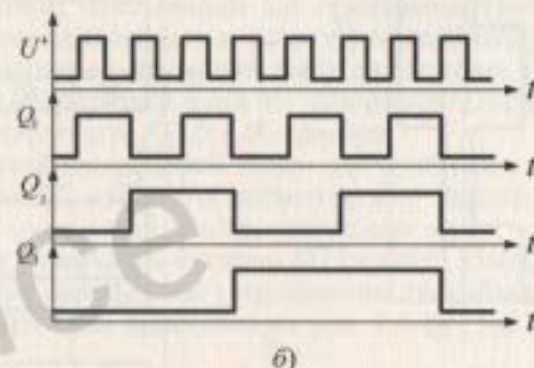
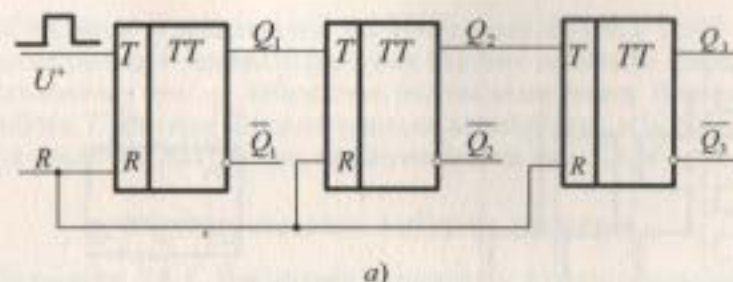


Рис. 5.5.3

Асинхронні віднімаючі лічильники на двохступінчатих T -тригерах будують аналогічно підсумовуючим, але з тією різницею, що сигнали переносу в лічильнику передаються асинхронно з інверсних виходів молодших розрядів на T -входи сусідніх старших. Схема такого лічильника на три розряди приведена на рис. 5.5.4а, а часова діаграма його роботи — на рис. 5.5.4б.

Реверсивний лічильник повинен суміщати принципи побудови підсумовуючого і віднімаючого лічильників. Схема його повинна задовольняти переключення із режиму додавання в режим віднімання і мати окремі установки лічильника в «0» і «1». Схема на три розряди, яка відповідає цим принципам, приведена на рис. 5.5.5.

У даній схемі міжрозрядні зв'язки комутуються за допомогою логічних елементів «2І – АБО». На виході цих логічних елементів виробляється сигнал T_i для лічильних входів старших розрядів.

$$T_i = Y_i \cdot Q_i \vee Y_i \cdot \bar{Q}_i, \quad i = 1, 2, 3, \dots, n.$$

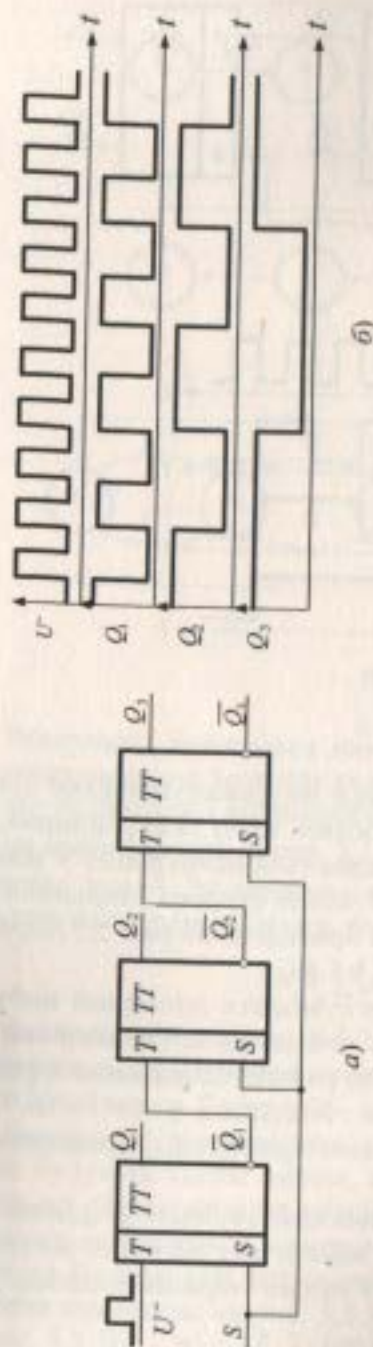


Рис. 5.5.4

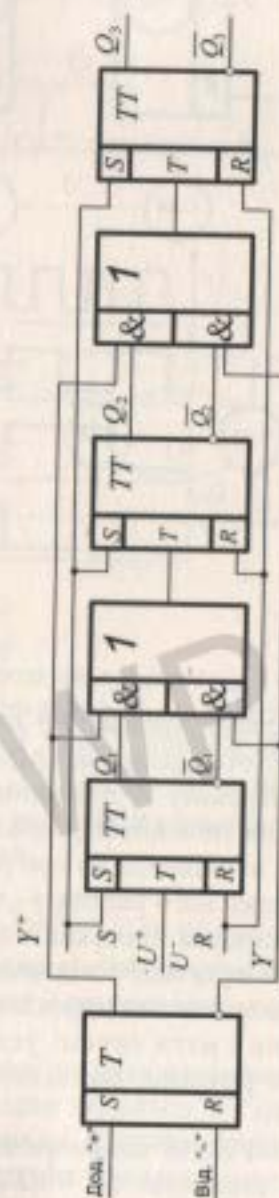


Рис. 5.5.5

Тобто, якщо управляючий RS -тригер перебуває у стані «1», то лічильник реалізує режим підрахунку вхідних імпульсів (додавання), а у протилежному разі — забезпечує режим віднімання. В обох режимах роботи T -тригери переключаються асинхронно, а їх робота описується часовими діаграмами, приведеними на рис. 5.5.3б і рис. 5.5.4б.

5.6. Схемотехніка побудови регістрів

Означення 5.6.1. Регістром називають функціональний пристрій комп'ютера, призначений для приймання, тимчасового зберігання, перетворення і видачі n -розрядного двійкового коду.

Регістр включає регулярний набір однотипних тригерів, у кожному із яких зберігається один біт інформації. Для регістрів найчастіше використовують RS , D і JK -тригери.

Регістр, який призначений тільки для приймання, зберігання і передачі інформації, називають елементарним. Регістр, в якому зберігання даних суміщається з мікроопераціями зсуву, називають зсувним. Регістри позначають буквами RG (register). Побудова елементарних регістрів з однофазним і парафазним способами запису інформації на RS -тригерах приведена на рис. 5.6.1а і 5.6.1б відповідно.

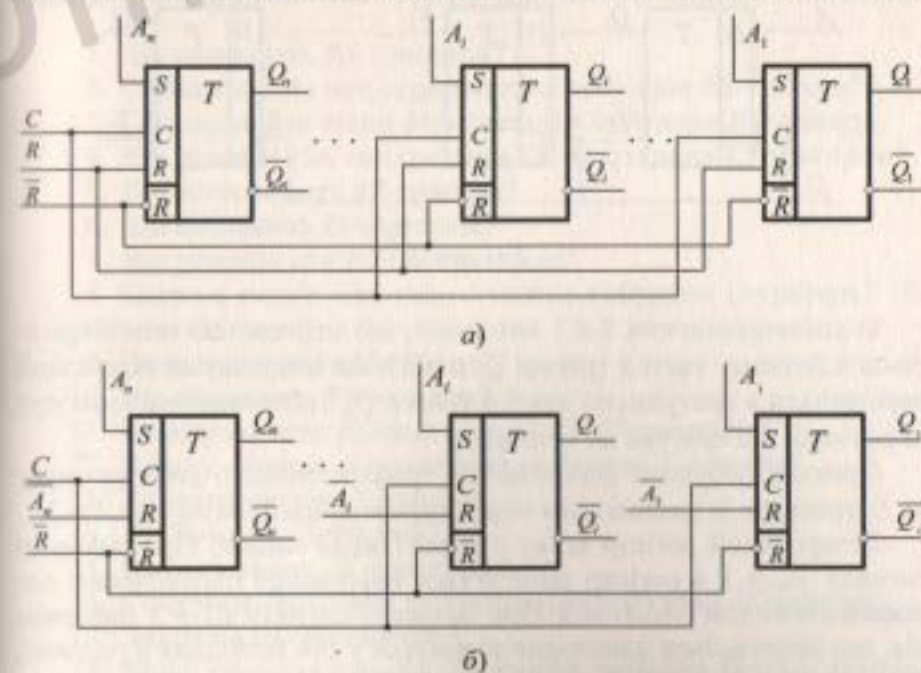


Рис. 5.6.1

Як впливає із рис. 5.6.1а, при однофазному запису інформації значення кожного слова $A_n \dots A_1$ подається по одній лінії зв'язку на вхід S RS -тригера, а після зчитування записаної інформації регістр повинен обнулитися за допомогою спільного $\bar{R}$ -входу. При парафазному способі запису значення кожного розряду слова передається за двома лініями зв'язку: пряме значення A_i поступає на вхід S , а інверсне $\bar{A}_i$ — на вхід R , рис. 5.6.1б. В цьому випадку не потрібна попередня установка регістра в стан «0», що дає можливість збільшити швидкість роботи регістра.

Зсувові регістри використовують у процесі виконання команд множення, ділення, нормалізації чисел, перетворення паралельного коду в послідовний і навпаки. Зсувові регістри проектують на двохступінчатих RS , JK або D -тригерах. На рис. 5.6.2 приведена побудова регістра зсуву, що складається з m послідовно з'єднаних D -тригерів, функції збудження яких мають вигляд

$$D_i = A, D_i = Q_{i-1}, i = 2, 3, \dots, m. \quad (5.6.1)$$

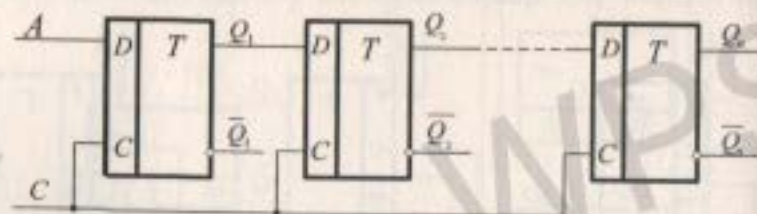


Рис. 5.6.2

Із співвідношення 5.6.1 випливає, що інформація яка зберігається в деякому такті в тригері Q_{i-1} , під дією імпульсу синхронізації передається в наступному такті в тригер Q_i , тобто відбувається зсув інформації від тригера до тригера.

Приклад побудови реверсивного трьохрозрядного регістру зсуву на D -тригерах із динамічним управлінням приведена на рис. 5.6.3.

Реверсивний регістр зсуву працює таким чином. При значенні сигналу $Y_m = 1$ в регістр записується інформація паралельним однофазним кодом ($A_3 A_2 A_1$). При значенні сигналу $R_n = 1$ інформація, що зберігається одночасно рухається у бік молодшого розряду, при цьому розряд тригера Q_3 обнуляється. При значенні сигналу

$L_n = 1$ інформація в регістрі одночасно рухається у бік старших розрядів, при цьому розряд Q_1 обнуляється. Запис і зсув інформації відбувається за переднім фронтом імпульсу (динамічний вхід CD -тригера).

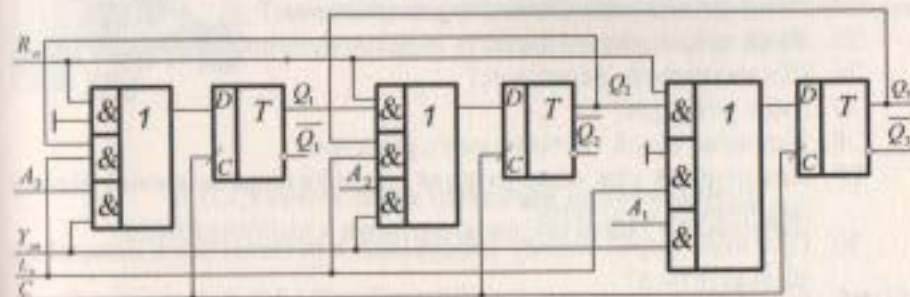


Рис. 5.6.3



Контрольні запитання

1. Що називають RS -тригером?
2. Скільки етапів має схемотехніка побудови RS -тригера?
3. Сформулюйте етапи схемотехніки побудови RS -тригера.
4. Яка різниця між синхронним і асинхронним RS -тригером?
5. Де застосовують RS -тригери?
6. Що називають D -тригером?
7. Яка різниця між RS і D -тригером?
8. Скільки етапів має схемотехніка побудови D -тригера? Назвіть їх.
9. Де застосовують D -тригери?
10. Що називають T -тригером?
11. В чому полягає різниця між RS , D і T -тригером?
12. В чому полягає схемотехніка побудови T -тригера?
13. Де застосовують T -тригер?
14. Які бувають T -тригери?
15. Що називають JK -тригером?
16. Сформулюйте етапи схемотехніки побудови JK -тригера.
17. Які бувають JK -тригери?
18. Чому двохступінчатий JK -тригер знайшов більше застосування, ніж одноступінчатий?

19. Що називають лічильником?
20. На яких тригерах проектують лічильники?
21. Які є лічильники і чим вони характеризуються?
22. За якими ознаками класифікують лічильники?
23. Що розуміють під модулем і ємністю лічильника?
24. Який лічильник називають реверсивним?
25. Який лічильник називають підсумовуючим і віднімаючим?
26. Що називають регістром?
27. Які є регістри?
28. Для яких цілей застосовують регістри?
29. Яка різниця між однофазною і парафазною формами запису інформації в регістрі?
30. При якій формі запису інформації в регістрі його швидкість збільшується?



Коментарі. В даному розділі для логіки побудови RS , D , T і JK -тригерів використані матеріали літератури [8, 11, 14], а логіка побудови лічильників і регістрів впливає з [1, 2, 22, 23].



Розділ 6

СХЕМОТЕХНІКА ПОБУДОВИ КОМП'ЮТЕРНИХ ПРИСТРОЇВ

6.1. Схемотехніка побудови одновихідних комбінаційних пристроїв на логічних елементах

Означення 6.1.1. Комбінаційним називають пристрій, вихідний сигнал якого визначається входним набором його змінних у даний момент часу.

Такі схеми не «пам'ятають» значення входних наборів у попередні моменти часу, а тому їх часто називають пристроями без пам'яті або одноклітковими схемами. При наявності одного виходу комбінаційні схеми описуються одним рівнянням (функцією) виду

$$y = f(x_1, x_2, \dots, x_n), \quad (6.1.1)$$

де x_i та y можуть прийняти лише значення логічного «0» або логічної «1».

Рівняння 6.1.1 отримують, як правило, за допомогою таблиці істинності, а форму цього рівняння називають ДДНФ, § 1.3. Інколи таке рівняння можна отримати на основі логіки роботи об'єкта.

За отриманим тим чи іншим способом рівнянням необхідно розробити схему, яка б реалізувала даний алгоритм. Методи проектування залежать від того, на якій елементарній базі буде реалізована дана схема. Тому логіка розробки комбінаційних схем на елементах логіки Буля має такі кроки.

1. За допомогою таблиці істинності або таблиці функціонування, або іншим способом описують алгоритм роботи необхідної схеми управління.

2. За таблицею функціонування або таблицею істинності визначають логічне рівняння для схеми управління.

3. Виконують мінімізацію логічного рівняння одним із методів, наведених у розділі 3.

4. При необхідності перетворюють мінімізоване логічне рівняння до певного базису елементів, на якому відбуватиметься побудова схеми.

5. Вибирають логічні елементи і будують принципіальну схему, яка відтворює отримане логічне рівняння.

Логіку побудови одновихідної комбінаційної схеми розглянемо на конкретних прикладах.

Приклад 6.1.1. У базисі Шеффера побудувати одновихідну комбінаційну схему, робота якої описується такою таблицею істинності, табл. 6.1.1.

Таблиця 6.1.1

x_1	x_2	x_3	y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1

x_1	x_2	x_3	y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Розв'язання. Використовуючи табл. 6.1.1, знаходимо логічне рівняння роботи схеми в ДДНФ, крок 2:

$$y = \bar{x}_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \vee \bar{x}_1 \cdot \bar{x}_2 \cdot x_3 \vee x_1 \cdot \bar{x}_2 \cdot x_3 \vee x_1 \cdot x_2 \cdot \bar{x}_3 \vee x_1 \cdot x_2 \cdot x_3.$$

На третьому кроці виконуємо мінімізацію отриманого логічного рівняння методом Карно:

		$x_2 x_3$			
x_1		00	01	11	10
	0			1	1
	1		1	1	1

$$y = x_2 \vee x_1 \cdot x_3. \quad (6.1.2)$$

Отримане рівняння 6.1.2 перетворюємо у базис Шеффера, крок 4.

$$y = x_2 \vee x_1 \cdot x_3 = \overline{\overline{x_2} \cdot \overline{x_1 \cdot x_3}}. \quad (6.1.3)$$

Користуючись рівнянням 6.1.3, а також елементами «І-НІ», які відповідають базису Шеффера, будемо одновихідну комбінаційну схему, рис. 6.1.1, крок 5.

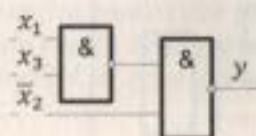


Рис. 6.1.1



Примітка. Вибір логічних елементів означає, перш за все, вибір серії мікросхем. Існують два основних класи мікросхем. Це мікросхеми на основі транзисторно-транзисторної логіки (ТТЛ) і мікросхеми на основі МОП-структури.

Реальні елементи обох класів мають також обмежену кількість входів, часову затримку і відповідні коефіцієнти розгалуження. При проектуванні схем конкретні параметри елементів у даному навчальному посібнику не розглядаються.

Приклад 6.1.2. У базисі Пірса побудувати одновихідну комбінаційну схему автомата, логіка роботи якого описується такою функцією

$$y = x_1 \cdot \bar{x}_2 \vee x_3 \cdot x_4 \vee \bar{x}_1 \cdot x_2 \cdot x_3. \quad (6.1.4)$$

Розв'язання. Мінімізацію заданої функції виконати неможливо, так як тоді два із трьох мінтерми (елементарні кон'юнкції) не відрізняються один від одного тільки однією змінною, тобто при винесенні за дужки загальних змінних неможливо отримати вираз $x \vee \bar{x} = 1$. Тому, користуючись кроком чотири, перетворимо рівняння 6.1.4 у базис Пірса, в результаті чого отримаємо

$$y = \overline{\overline{\bar{x}_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee x_1 \vee \bar{x}_2 \vee \bar{x}_3}}. \quad (6.1.5)$$

Вибір логічних елементів необхідно робити так само, як і у прикладі 6.1.1. Для побудови схеми у базисі Пірса використаємо рівняння 6.1.5, логічні зв'язки якого реалізуємо на елементах «АБО-НІ», рис. 6.1.2.

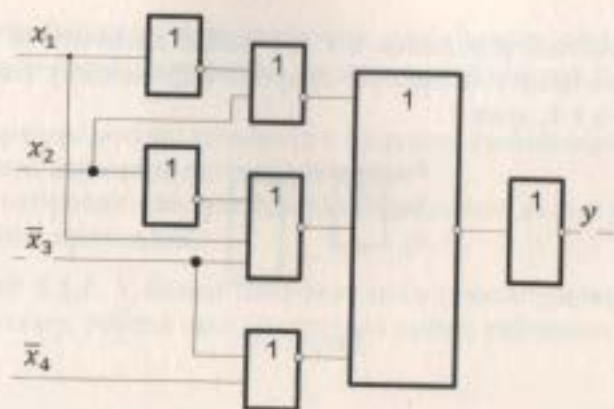


Рис. 6.1.2

Приклад 6.1.3. У базисі елементів «І», «АБО», «НІ» побудувати одновихідну комбінаційну схему, робота якої описується такою таблицею істинності, табл. 6.1.2.

Таблиця 6.1.2

x_1	x_2	x_3	y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1

x_1	x_2	x_3	y
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Розв'язання. Використовуючи табл. 6.1.2, знаходимо логічне рівняння роботи схеми в ДДНФ

$$y = \bar{x}_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \vee \bar{x}_1 \cdot \bar{x}_2 \cdot x_3 \vee \bar{x}_1 \cdot x_2 \cdot x_3 \vee x_1 \cdot \bar{x}_2 \cdot x_3.$$

Виконаємо мінімізацію отриманого логічного рівняння методом Карно,

		$x_2 x_3$			
x_1		00	01	11	10
	0	1	1	1	
	1		1		

в результаті чого отримаємо

$$y = \bar{x}_1 \cdot \bar{x}_2 \vee \bar{x}_1 \cdot x_3 \vee \bar{x}_2 \cdot x_3. \quad (6.1.6)$$

Вибір логічних елементів необхідно робити так же, як і у прикладі 6.1.1. Для побудови схеми використаємо рівняння 6.1.6, логічні зв'язки якого реалізуємо за допомогою елементів «І», «АБО», «НІ», рис. 6.1.3.

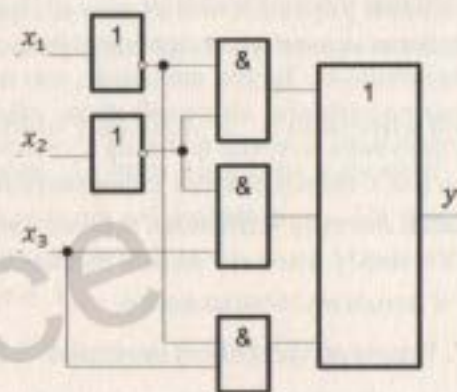


Рис. 6.1.3

6.2. Схемотехніка побудови одновихідних комбінаційних схем на мультиплексорах

В § 3.3 було показано, що в мультиплексорі номер набору на управляючому вході дорівнює номеру інформаційного входу x_i , який підключений до виходу D . Тоді загальне рівняння роботи мультиплексора матиме такий вигляд

$$D = \bigvee_{i=0}^{2^n-1} A_i \cdot b_i,$$

де A_i — номер вхідного набору на управляючих входах;

b_i — змінна на i -му інформаційному вході;

D — вихідне значення функції мультиплексора.

У подальшому вихідне значення функції мультиплексора D будемо позначати через y .

Із рівняння 6.2.1 випливає, що мультиплексор розкладає логічну функцію за n змінними. Подібні властивості мультиплексорів і використовують при синтезі комбінаційних схем.

Логіка побудови комбінаційних схем на мультиплексорах залежить від кількості змінних у функції та від кількості управляючих входів мультиплексора. Тому є декілька варіантів логіки побудови комбінаційних схем на мультиплексорах.

Варіант перший. У даному варіанті кількість змінних у функції m дорівнює кількості управляючих входів n . При цьому розклад функції з m змінними за n змінними дає мінтерми, які можуть бути нульовими або одиночними. Звідси випливає, що на входи мультиплексора необхідно подати або логічний нуль, або логічну одиницю. Оскільки розглядувана логічна функція є сумою одиничних мінтермів, то числа в ній є номерами тих інформаційних входів b_i , на які необхідно подати логічну одиницю, а решту входів з'єднати з логічним нулем. У такому випадку вхідні змінні x_i необхідно подати на управляючі входи мультиплексора.

Приклад 6.2.1. Реалізувати логічну функцію

$$y = x_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \vee \bar{x}_1 \cdot \bar{x}_2 \cdot x_3 \vee \bar{x}_1 \cdot x_2 \cdot \bar{x}_3.$$

на мультиплексорі.

Розв'язання. У даній функції кількість змінних $m = 3$, тому необхідно застосувати мультиплексор для її реалізації з кількістю управляючих входів $n = 3$.

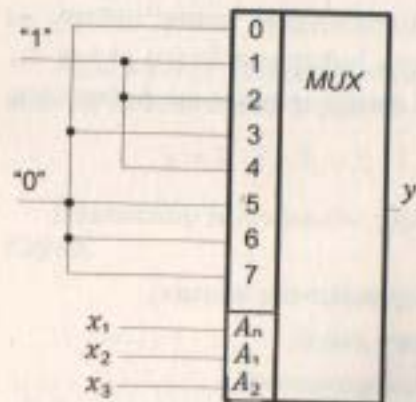


Рис. 6.2.1

Із логічного рівняння прикладу знаходимо десяткові числа, яким відповідають мінтерми в функції $y = \vee(1, 2, 4)$. Тобто, на інформаційні входи 1, 2, 4 мультиплексора необхідно подати логічні одиниці, а на решту — нулі.

Тоді схема реалізації заданої функції на мультиплексорі при подачі на його управляючі входи змінних x_1, x_2, x_3 матиме вигляд, приведений на рис. 6.2.1.

Варіант другий. У даному варіанті кількість змінних $m = n + 1$.

Це засвідчує те, що кількість змінних у функції більша за кількість управляючих входів мультиплексора. В даному випадку задану функцію можна розкласти за будь-якою змінною, але найбільш доцільно зробити це або за старшою, або за молодшою змінною. Якщо функцію розкласти за старшою змінною, то будемо мати

$$f(x_1, x_2, \dots, x_m) = \bar{x}_1 \cdot f_1(0, x_2, \dots, x_m) \vee x_1 \cdot f_2(1, x_2, \dots, x_m).$$

Якщо функції f_1 і f_2 мають однакові мінтерми, то це означає, що на відповідний b_i вхід мультиплексора необхідно подати логічну одиницю.

Решта мінтермів у функції f_1 є номерами інформаційних входів, на які подають $\bar{x}_1$. Інші мінтерми в функції f_2 є номерами інформаційних входів, на які подають x_1 . На решту входів мультиплексора подають логічний нуль, а на управляючі входи мультиплексора — змінні $x_2, \dots, x_m$.

Приклад 6.2.2. Реалізувати логічну функцію

$$y = \bar{x}_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \cdot x_4 \vee \bar{x}_1 \cdot \bar{x}_2 \cdot x_3 \cdot x_4 \vee \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 \cdot x_4 \vee \bar{x}_1 \cdot x_2 \cdot x_3 \cdot \bar{x}_4 \vee x_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \cdot x_4 \vee x_1 \cdot \bar{x}_2 \cdot x_3 \cdot x_4 \vee x_1 \cdot x_2 \cdot \bar{x}_3 \cdot \bar{x}_4 \vee x_1 \cdot x_2 \cdot x_3 \cdot x_4$$

на мультиплексорі, який має три управляючі входи.

Розв'язання. Так як мультиплексор має три управляючі входи, тобто $n = 3$, а $m = 4$, то для розв'язку прикладу необхідно використати другий варіант. Для наочності і спрощення рішення перетворимо мінтерми заданої функції у десяткові числа

$$y = \vee(1, 3, 5, 6, 9, 11, 12, 15).$$

Розкладемо функцію за старшою змінною, в результаті чого рівняння матиме вигляд $y = \bar{x}_1 \cdot [\vee(1, 3, 5, 6)] \vee x_1 \cdot [\vee(9, 11, 12, 15)]$.

В обох частинах розкладання є спільні числа 1 та 3, тому:

$$b_1 = b_3 = 1; b_5 = b_6 = \bar{x}_1; b_4 = b_7 = x_1.$$

На решту входів мультиплексора необхідно подати логічний нуль, а на його управляючі входи змінні x_2, x_3, x_4 . Виходячи із цього,

реалізація заданої логічної функції на мультиплексорі з трьома управляючими входами матиме вигляд, приведений на рис. 6.2.2.

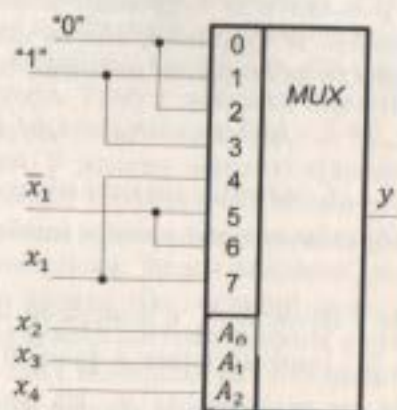


Рис. 6.2.2

При подальшому збільшенні кількості змінних у логічній функції необхідно зробити її розкладання за двома і т.д. змінними, із наступним використанням розглянутої вище логіки побудови одновихідних комбінаційних схем на мультиплексорах.

6.3. Схемотехніка побудови багатовихідних комбінаційних схем на логічних елементах

Означення 6.3.1. Багатовихідною комбінаційною схемою називають схему, яка управляє декількома елементами (механізмами) одночасно.

Вона задається системою булевих функцій

$$y_1 = f_1(x_1, x_2, \dots, x_n),$$

$$y_2 = f_2(x_1, x_2, \dots, x_n),$$

$$\dots$$

$$y_m = f_m(x_1, x_2, \dots, x_n).$$

Особливістю цих рівнянь є те, що вони реалізують різні функції від одних і тих же змінних.

Початковий стан проектування багатовихідних схем такий же, як і при проектуванні одновихідних (за допомогою таблиці істинності або іншим шляхом отримують конкретні функції для кожного виходу).

Кожна функція, незалежно одна від одної, ймовірно реалізується окремо як одновихідна функція. Але така система може бути не оптимальною через те, що окремі рівняння можуть мати деяку кількість однакових мінтермів, які доцільно реалізувати на спільних елементах. Тому пошук спільних імплікант і становить суть методу проектування подібних схем.

Даний метод оснований на пошуку спільних імплікант заданих функцій шляхом їх спільної мінімізації. Логіка послідовності цього методу має такі кроки.

1. Знайти прості імпліканти кожної із функцій.
2. Знайти прості імпліканти добутку всіх можливих пар заданих функцій

$$f_1 \cdot f_2; f_1 \cdot f_3; \dots; f_1 \cdot f_m; f_2 \cdot f_3; f_2 \cdot f_4; \dots; f_2 \cdot f_m; \dots; f_{m-1} \cdot f_m.$$

3. Знайти прості імпліканти добутку всіх можливих поєднань трьох, чотирьох і т.п. функцій. Останнім буде знайдено прості імпліканти добутку всіх функцій. Під добутком функцій розуміють їх спільні мінтерми.

4. Залишити серед отриманих однакових імплікант тільки одні і позначити буквами.

5. Серед отриманих імплікантів методом Квайна вилучити зайві.

6. Розподілити отримані імпліканти за функціями таким чином, щоб кожна функція була сумою певної кількості імплікант.

У подальшому проектування багатовихідної комбінаційної схеми виконують аналогічно проектуванню одновихідних комбінаційних схем.

Приклад 6.3.1. Побудувати двохвихідну комбінаційну схему, яка задана системою функцій у вигляді десяткових чисел

$$y_1 = \vee (6, 7, 8, 9, 10, 11, 13, 14, 15),$$

$$y_2 = \vee (1, 3, 6, 7, 8, 9, 10, 11). \quad (6.3.1)$$

Розв'язання. За допомогою карт Карно, рис. 6.3.1, знайдемо прості імпліканти кожної із функцій, відмічаючи мінтерми, з яких вони отримані.

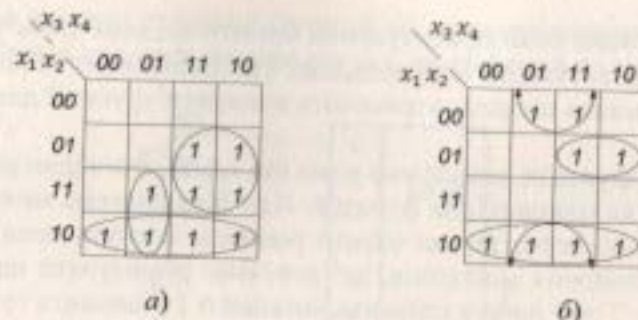


Рис. 6.3.1

Із рис. 6.3.1а маємо

$$y_1 = x_1 \cdot \bar{x}_2 (8, 9, 10, 11) \vee x_2 \cdot x_3 (6, 7, 14, 15) \vee x_1 \cdot \bar{x}_3 \cdot x_4 (9, 13);$$

$$y_1 = x_1 \cdot \bar{x}_2 (8, 9, 10, 11) \vee \bar{x}_2 \cdot x_4 (1, 3, 9, 11) \vee \bar{x}_1 \cdot x_2 \cdot x_3 (6, 7).$$

Користуючись рис. 6.3.1, знайдемо спільні імпліканти добутку функцій y_1 та y_2 . Відповідна карта Карно для добутку цих функцій приведена на рис. 6.3.2.

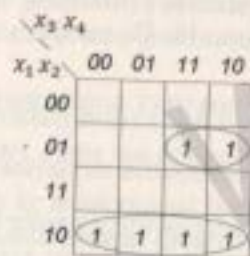


Рис. 6.3.2

Із рис. 6.3.2 маємо

$$y_{12} = x_1 \cdot \bar{x}_2 (8, 9, 10, 11) \vee \bar{x}_1 \cdot x_2 \cdot x_3 (6, 7).$$

Позначимо кожний імплікант латинськими буквами, пропускаючи ті з них, які вже зустрічались раніше

$$A = x_1 \cdot \bar{x}_2 (8, 9, 10, 11);$$

$$B = x_2 \cdot x_3 (6, 7, 14, 15);$$

$$C = x_1 \cdot \bar{x}_3 \cdot x_4 (9, 13);$$

$$D = \bar{x}_2 \cdot x_4 (1, 3, 9, 11);$$

$$E = \bar{x}_1 \cdot x_2 \cdot x_3 (6, 7).$$

Будуємо таблицю Квайна, в якій стовпчики відповідають мінтермам функцій, а рядки — імплікантам, табл. 6.3.1.

Таблиця 6.3.1

Імпліканти	Мінтерми функцій																
	y_1										y_2						
	6	7	8	9	10	11	13	14	15	1	3	6	7	8	9	10	11
A			*	*	*	*								*	*	*	*
B	*	*						*	*								
C				*			*										
D										*	*				*		*
E												*	*				

Заповнення таблиці здійснюють таким чином. Для кожного відібраного імпліканта зірочку ставлять у стовпчику, номер якого є в списку мінтермів функцій цього імпліканта. Так, наприклад, імплікант A належить функції y_1 і y_2 , тому зірочку проставляють у стовпчиках 8, 9, 10, 11 першої і другої функцій. Аналогічно заповнюють зірочками таблицю і для решти імплікант.

У таблиці Квайна, в першу чергу, розглянемо стовпчики, сума зірочок в яких дорівнює одиниці. Так, наприклад, для імпліканти A першим розглядаємо стовпчик 8 функції y_1 , так як він здатний забезпечити одиничний сигнал для даної функції.

Решту стовпчиків даної імпліканти викреслюємо як для функції y_1 , так і для функції y_2 . Імпліканта B у кожному стовпчику теж має одну зірочку, тому можна розглянути стовпчик 6, а решту викреслити. Для імпліканти C першим необхідно розглянути стовпчик 13, так як він має одну зірочку, а решту викреслити, а для імплікант D і E — стовпчики 1 і 6 функції y_2 відповідно, а решту викреслити.

Таким чином для реалізації схеми необхідно використати п'ять імплікант — A, B, C, D, E. Зайвих імплікант при мінімізації не виявлено. Тепер необхідно задані імпліканти розподілити між функ-

ціями. Із табл. 6.3.1 випливає, що імплікант A належить обом функціям, імпліканти B і C тільки функції y_1 , а D і E — лише функції y_2 . Функції, як сума відповідних імплікант, матимуть вигляд

$$y_1 = A \vee B \vee C = x_1 \cdot \bar{x}_2 \vee x_2 \cdot x_3 \vee x_1 \cdot \bar{x}_3 \cdot x_4;$$

$$y_2 = A \vee D \vee E = x_1 \cdot \bar{x}_2 \vee \bar{x}_2 \cdot x_4 \vee \bar{x}_1 \cdot x_2 \cdot x_3.$$

Побудова комбінаційної схеми з використанням рівнянь 6.3.2 відбувається аналогічно описаному в § 6.1. Дана схема на елементах логіки Буля має вигляд, рис. 6.3.3.

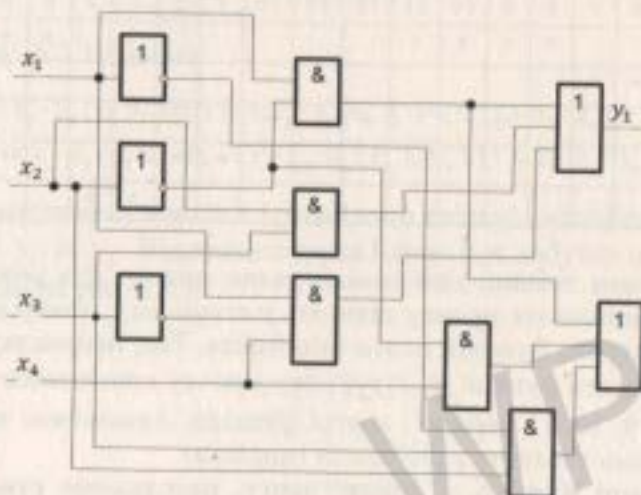


Рис. 6.3.3

6.4. Схемотехніка побудови багатовихідних комбінаційних схем на дешифраторах

Значна кількість кроків у побудові схем робить класичний метод проєктування багатовихідних комбінаційних схем інколи незручним для застосування. Обминути цей метод можна шляхом застосування дешифраторів. Із § 3.1 випливає, що кожний вихід дешифратора відповідає певному вхідному набору змінних, тобто певному мінтерму. Оскільки кожна функція є сумою одиничних мінтермів, то для її реалізації достатньо об'єднати елементом «АБО» ті виходи дешифратора, які відповідають одиничним мінтермам. Невикористані набори, якщо вони є, слід віднести до нульових.

Приклад 6.4.1 Побудувати чотирьохвихідну комбінаційну схему, яка задана системою функцій у вигляді десяткових чисел

$$y_1 = \vee (0, 1, 3, 4, 5);$$

$$y_2 = \vee (1, 2, 4, 5, 6);$$

$$y_3 = \vee (0, 2, 3, 6);$$

$$y_4 = \vee (0, 1, 3, 5, 6).$$

Розв'язання. Використовуючи § 3.1, будемо двійковий дешифратор, який забезпечує на своїх виходах реалізацію чотирьох функцій у вигляді десяткових чисел від 0 до 6. Тобто дешифратор повинен мати сім виходів. Згідно з даними § 3.1, дешифратор за кількістю входів і виходів зв'язаний співвідношенням $m = 2^n$, де n — кількість входів, а m — кількість виходів. У такому випадку для побудови двійкового дешифратора на сім виходів необхідно мати $n = \lceil \log_2 7 \rceil = \lceil 2,807 \rceil = 3$.

Виходячи з цього, а також використовуючи дані § 3.1, будемо чотирьохвихідну комбінаційну схему реалізацій функцій y_1, y_2, y_3, y_4 на дешифраторі і елементах «АБО», рис. 6.4.1.

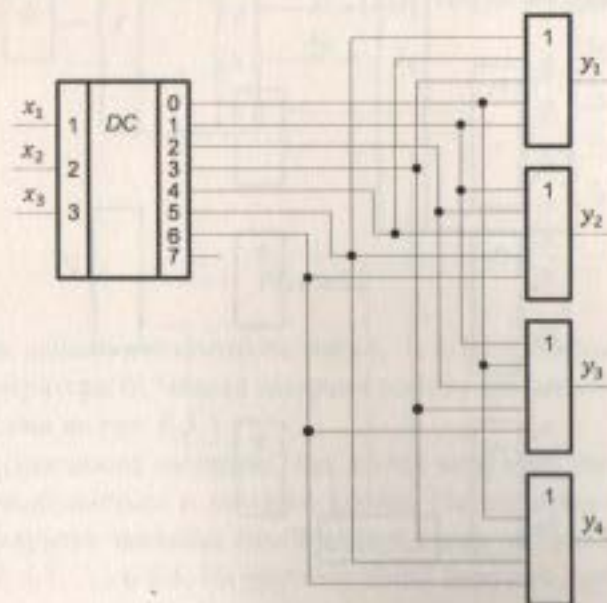


Рис. 6.4.1

Із рис. 6.4.1 випливає, що невикористаним набором є сьомий вихід дешифратора, який не бере участі в жодній із заданих за умовою функцій.

При збільшенні кількості змінних у функціях збільшують кількість дешифраторів, що приводить до утворення дешифраторного дерева.

6.5. Схемотехніка побудови часових логічних схем

Робота часової логічної функції в загальному вигляді описується рівнянням виду $y = \varphi(x_1, x_2, \dots, x_n, t), t = 1, 2, \dots, k$.

Якщо маємо різні логічні функції в різні моменти часу, то їх можна записати у такому вигляді:

$$y = \varphi_1 \cdot t_1 \vee \varphi_2 \cdot t_2 \vee \dots \vee \varphi_n \cdot t_k. \quad (6.5.1)$$

де φ_i — значення i -ї функції, яке визначає i -у компоненту в необхідній послідовності в момент часу t_i .

Структурна схема часової логічної функції, яка реалізує рівняння 6.5.1, наведена на рис. 6.5.1.

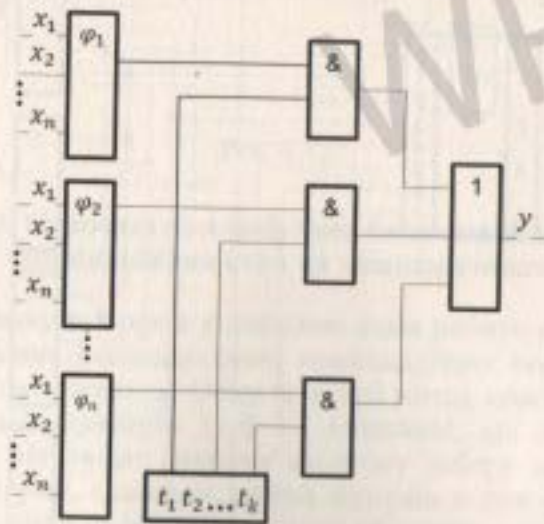


Рис. 6.5.1

Блоки $\varphi_1, \varphi_2, \dots, \varphi_n$ реалізують функції $\varphi_1, \varphi_2, \dots, \varphi_n$, причому кожен з них реалізується звичайною вихідною комбінаційною схемою. Виходи блоків подаються на ключі, які реалізовані на елементах «І». Якщо на виході датчика часу буде організована послідовна видача сигналів $t_1, t_2, \dots, t_n$, то на виходах схем «І» будуть з'являтися послідовно в часі сигнали функцій $\varphi_1, \varphi_2, \dots, \varphi_n$. При управлінні одним механізмом ці сигнали необхідно об'єднати елементом «АБО», але якщо необхідно здійснювати управління декількома механізмами, то цей елемент є зайвим.

Датчик часу може бути реалізований на дешифраторі та лічильнику, до якого підключають генератор G . Схема такого датчика часу на вісім затримок приведена на рис. 6.5.2.

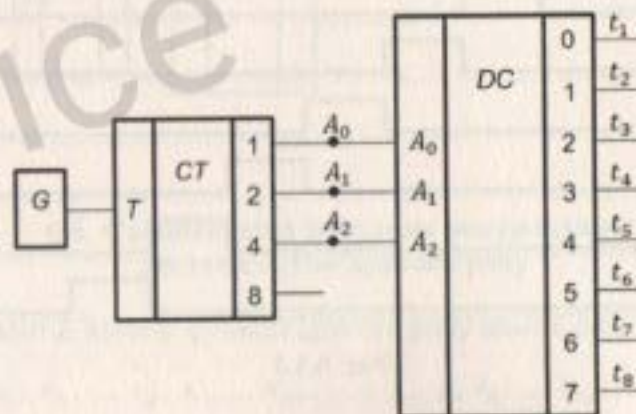


Рис. 6.5.2

В таких схемах дискретність часу $t_1 \dots t_k$ визначається частотою роботи генератора G . Часова діаграма роботи датчика часу для рис. 6.5.2 наведена на рис. 6.5.3.

Із вищесказаного випливає, що логіка побудови часових логічних схем відбувається в декілька кроків. На першому кроці необхідно побудувати звичайні комбінаційні схеми за правилами, наведеними в § 6.1, ..., § 6.4. На другому кроці будують датчик часу, на третьому — ключі, а на четвертому — часову логічну схему, вико-

ристовуюючи перший, другий і третій кроки, а також елементи булевих функцій, наведених у § 2.6.

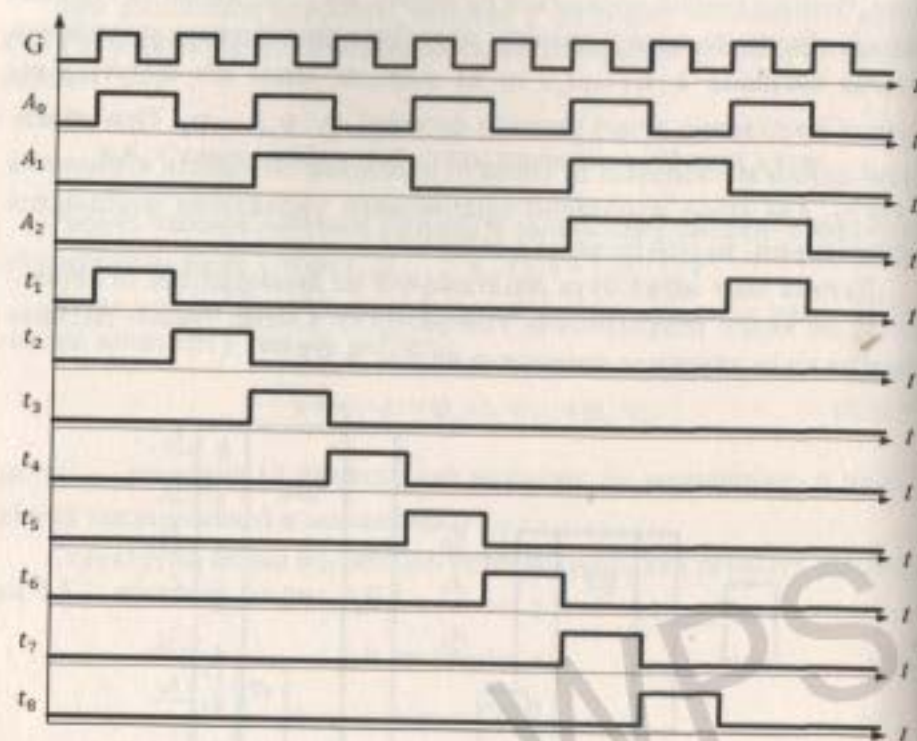


Рис. 6.5.3

Приклад 6.5.1. Побудувати часову логічну схему для функції

$$y_i = x_1 \cdot \bar{x}_2 \cdot t_1 \vee \bar{x}_1 \cdot x_2 \cdot t_2 \vee x_1 \cdot t_3 \vee x_2 \cdot t_4.$$

Розв'язання. Для реалізації заданого рівняння на першому кроці будемо звичайні комбінаційні схеми мінтермів $x_1 \cdot \bar{x}_2$ і $\bar{x}_1 \cdot x_2$ на другому — датчик часу на чотири затримки, на третьому — ключі з використанням елементів «Д», а на четвертому — саму часову логічну схему, наведену на рис. 6.5.4.

Багатовихідні часові логічні схеми будуються аналогічно одновихідним, але тільки без елемента «АБО».

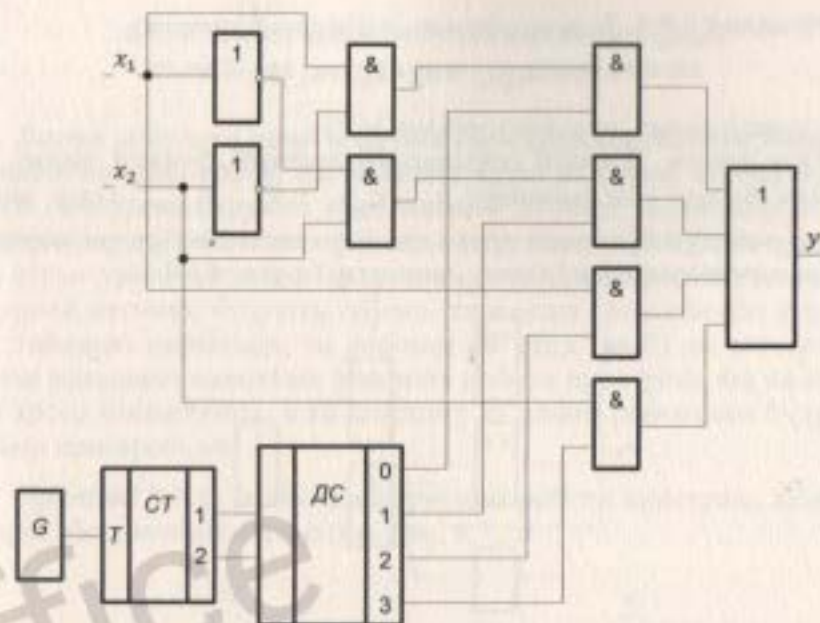


Рис. 6.5.4

6.6. Схемотехніка побудови рекурентних логічних схем другого роду

Рекурентні логічні функції другого роду мають вигляд

$$y = \Phi(x_{1t}, x_{2t}, \dots, x_{mt}, x_{1(t-1)}, x_{2(t-1)}, \dots, x_{m(t-1)}, x_{1(t-r)}, x_{2(t-r)}, \dots, x_{n(t-r)}).$$

Це функції, у яких між змінними, наприклад, x_{1t} і $x_{1(t-1)}$, є часовий зсув.

Такі функції не мінімізують, тому що значення однієї і тієї ж змінної залежить від часу.

Логіка побудови рекурентних логічних схем другого роду має такі кроки. На першому кроці будують рекурентну логічну функцію, яка описує роботу того чи іншого пристрою, а на другому — реалізують її, використовуючи § 6.1, ..., § 6.4, без урахування часових затримок. Побудова рекурентної логічної функції закінчується введенням між змінними часових затримок на третьому кроці. Часові затримки можна реалізувати на малогабаритних лініях затримки (МЛЗ), або D-тригерах, робота яких описана в § 5.2.

Приклад 6.6.1. За рекурентною логічною функцією

$$y = x_{1(t-1)} \vee x_u \cdot \bar{x}_{2t} \vee x_{2(t-2)}$$

побудувати схему, використовуючи МЛЗ.

Розв'язання. В даній рекурентній логічній функції часові затримки введені між змінними x_n , $x_{1(n-1)}$ і $\bar{x}_{2i}$, $x_{2(n-2)}$. Тому, використовуючи § 6.1, а також кроки два і три логіки побудови, отримаємо рекурентну логічну схему, наведену на рис. 6.6.1.

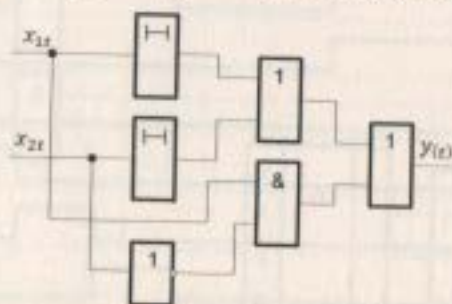


Рис. 6.6.1

Приклад 6.6.2. За рекурентною логічною функцією прикладу 6.6.1 побудувати схему, використовуючи в якості елементів часової затримки D -тригер.

Розв'язання. Побудова схеми відбувається аналогічно прикладу 6.6.1 з тією лише різницею, що замість МЛЗ вводимо синхронні *D*-тригери, § 5.2. Тоді отримаємо схему, яка реалізує рекурентну логічну функцію прикладу 6.6.1, рис. 6.6.2.

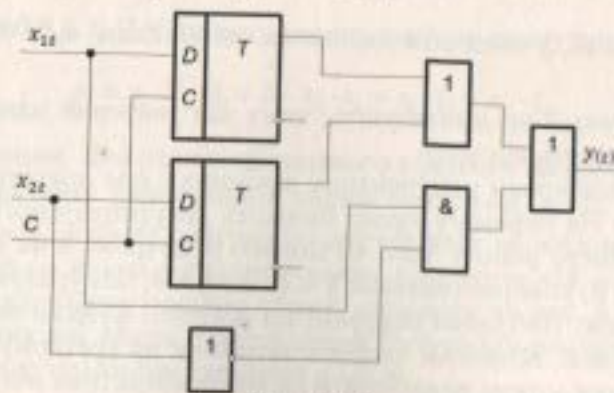
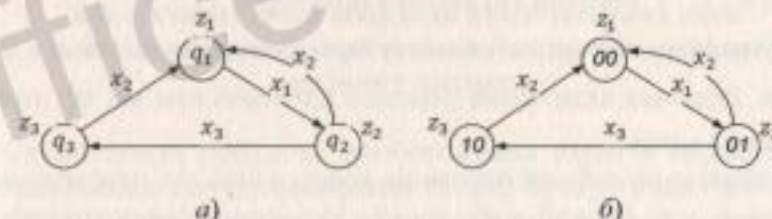


Рис. 6.6.2

6.7. Схемотехніка побудови комп'ютерних схем із застосуванням теорії автоматів

Логіка побудови комп'ютерних схем із застосуванням теорії автоматів має такі кроки. На першому кроці на основі алгоритму роботи пристрою будують абстрактний автомат, який кодується на другому кроці двійковим нормальним кодом, отримуючи при цьому структурний автомат. На третьому кроці, використовуючи структурний автомат, будують таблиці переходів і виходів або відмічену таблицю переходів, за допомогою яких (якої) на четвертому кроці знаходять канонічні рівняння роботи пристрою, які на п'ятому кроці мінімізують, а на шостому за даним рівнянням будують схему пристрою.

Приклад 6.7.1. Згідно з алгоритмом роботи пристрою, який заданий абстрактним автоматом, рис. 6.7.1а.



$q_j \backslash z_k$	z_1	z_2	z_3
$x_1 \backslash q_j$	00	01	10
x_1	01	-	-
x_2	-	00	00
x_3	-	10	-

Рис. 6.7.1

побудувати один із вузлів комп'ютерної схеми управління.

Розв'язання. Для забезпечення реалізації трьох станів абстрактного автомата q_1, q_2, q_3 необхідно в структурному автоматі мати згідно з формулою $n = \lceil \log_2 3 \rceil = \lceil 1,807 \rceil = 2$ два елементи пам'яті, які можуть задовільнити реалізацію чотирьох станів: 00, 01, 10, 11.

У нашому випадку для кодування станів абстрактного автомата використаємо кодові стани $q_1 \rightarrow 00$, $q_2 \rightarrow 01$, $q_3 \rightarrow 10$. У результаті цього структурний автомат матиме вигляд, рис. 6.7.1б. Використовуючи відмічену для автомата Мура таблицю переходів, рис. 6.7.1в, отримаємо канонічні рівняння роботи схеми пристрою, які матимуть такий вигляд:

$$z_1 = \bar{y}_1 \cdot \bar{y}_2; z_2 = \bar{y}_1 \cdot y_2; z_3 = y_1 \cdot \bar{y}_2;$$

$$\phi_1^1 = x_3; \phi_1^0 = x_2 \cdot \bar{y}_2; \phi_2^1 = x_1 \cdot \bar{y}_1; \phi_2^0 = x_3;$$

де ϕ_1^1 , ϕ_2^1 і ϕ_1^0 , ϕ_2^0 — функції включення і виключення відповідно до першого і другого елементів пам'яті структурного автомата;

y_1 , y_2 і $\bar{y}_1$, $\bar{y}_2$ — сигнали на виходах першого і другого елементів пам'яті, які є логічними сигналами «1» і «0» відповідно;

z_1 , z_2 , z_3 — сигнали управління пристроєм.

Функція ϕ_1 відповідає елементу кода, розміщеного зліва, а ϕ_2 — справа. Рівняння включення першого елемента пам'яті ϕ_1^1 отримують таким чином.

У відміченій таблиці переходів розглядають усі переходи кодових станів цієї функції з «0» до «1» під дією вхідних змінних. До кон'юнкції вхідних змінних також записують і змінну другого елемента пам'яті, якщо вона не міняє свій знак при цьому переході. Якщо цей перехід для функції ϕ_1^1 відбувається не один раз, а, наприклад, два, то знайдені кон'юнкції змінних об'єднують знаком диз'юнкції.

Рівняння виключення першого елемента пам'яті ϕ_1^0 отримують аналогічно описаному з тією лише різницею, що при цьому розглядають лише переходи із стану «1» до стану «0». Рівняння для функції ϕ_2^0 отримують аналогічно описаному для функції ϕ_1^0 .

Як видно із отриманих рівнянь, їх мінімізація не потрібна, тому переходимо до шостого кроку, де будемо схему пристрою. При побудові схеми використовуємо § 5.1, § 6.3. Наведена схема комп'ютерного пристрою має вигляд, рис. 6.7.2.

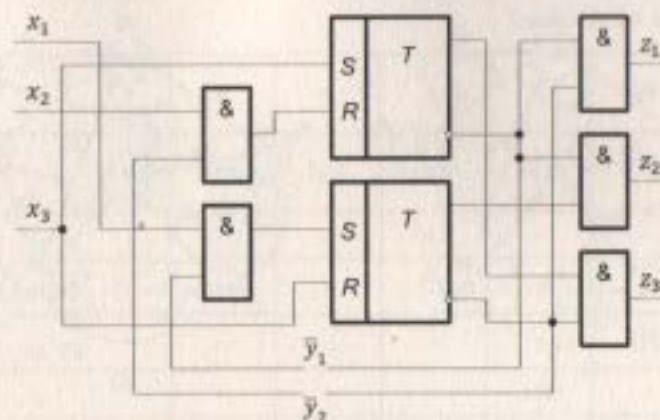


Рис. 6.7.2

6.8. Схемотехніка побудови комп'ютерних схем із застосуванням теорії автоматів і програмованих логічних матриць

Схемотехніка побудови комбінаційних схем із застосуванням програмованих логічних матриць (ПЛМ) була розглянута в § 5.5, а схемотехніка побудови комп'ютерних схем із використанням теорії автоматів в § 6.7. Об'єднавши ці дві схемотехніки, ми отримаємо таку схемотехніку побудови комп'ютерних схем із застосуванням теорії автоматів і ПЛМ. Вона складається із семи кроків. На першому кроці на основі заданого алгоритму роботи будують абстрактний автомат, який на другому кроці кодують двійковим нормальним кодом. На третьому кроці, використовуючи структурний автомат, отриманий на другому кроці, будують таблиці переходів і виходів, за допомогою яких (якої) на четвертому кроці знаходять канонічні рівняння роботи. Мінімізація їх відбувається на п'ятому кроці, а на шостому — вибір необхідної ПЛМ і її програмування. На сьомому кроці за вибраною і запрограмованою ПЛМ, а також за отриманими на другому кроці елементами пам'яті будують необхідну схему.

Приклад 6.8.1. За алгоритмом роботи пристрою, який заданий абстрактним автоматом, рис. 6.8.1а.

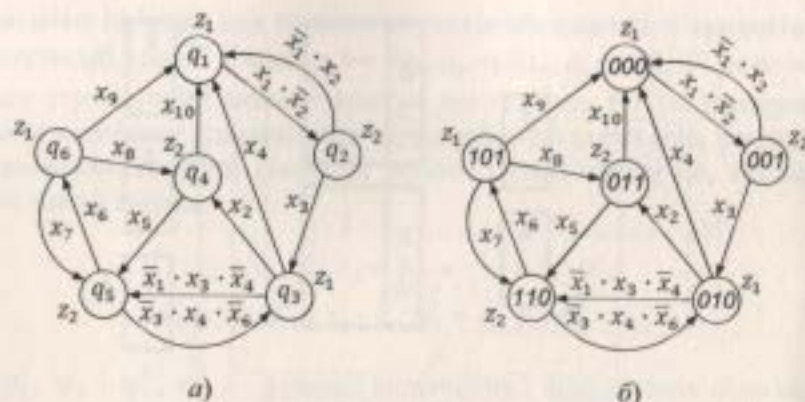


Рис. 6.8.1

Побудувати комп'ютерну схему управління з низьким рівнем активності для функцій z і високим — для решти функцій.

Розв'язання. Щоб забезпечити реалізацію шести станів абстрактного автомата $q_1, \dots, q_6$, необхідно в структурному автоматі мати згідно з формулою $n = \lceil \log_2 6 \rceil = 3$ три елементи пам'яті, які можуть задовільнити реалізацію восьми станів: 000, 001, 010, 011, 100, 101, 110, 111. У нашому випадку для кодування абстрактного автомата використаємо кодові стани: $q_1 \rightarrow 000$; $q_2 \rightarrow 001$; $q_3 \rightarrow 010$; $q_4 \rightarrow 011$; $q_5 \rightarrow 110$; $q_6 \rightarrow 101$. У результаті цього структурний автомат матиме такий вигляд, рис. 6.8.1б. Використовуючи відмічену для автомата Мура таблицю переходів, табл. 6.8.1.

Таблиця 6.8.1

$q_j \backslash z_k$	z_1	z_2	z_3	z_4	z_5	z_6
q_j	000	001	010	011	110	101
x_i						
$x_1 \cdot \bar{x}_2$	001	—	—	—	—	—
$\bar{x}_1 \cdot x_2$	—	000	—	—	—	—
x_3	—	010	—	—	—	—
x_4	—	—	000	—	—	—

Закінчення табл. 6.8.1

$q_j \backslash z_k$	z_1	z_2	z_3	z_4	z_5	z_6
q_j	000	001	010	011	110	101
x_i						
x_1	—	—	011	—	—	—
$\bar{x}_1 \cdot x_3 \cdot \bar{x}_4$	—	—	110	—	—	—
$\bar{x}_3 \cdot x_4 \cdot \bar{x}_6$	—	—	—	—	010	—
x_5	—	—	—	110	—	—
x_6	—	—	—	—	—	110
x_7	—	—	—	—	—	110
x_8	—	—	—	—	—	011
x_9	—	—	—	—	—	000
x_{10}	—	—	—	000	—	—

отримаємо канонічні рівняння роботи схеми, які матимуть такий вигляд:

$$z_1 = \bar{y}_1 \cdot \bar{y}_2 \cdot \bar{y}_3 \vee \bar{y}_1 \cdot y_2 \cdot \bar{y}_3 \vee y_1 \cdot \bar{y}_2 \cdot y_3;$$

$$z_2 = \bar{y}_1 \cdot \bar{y}_2 \cdot y_3 \vee \bar{y}_1 \cdot y_2 \cdot y_3 \vee y_1 \cdot \bar{y}_2 \cdot \bar{y}_3;$$

$$\varphi_1^1 = \bar{x}_1 \cdot x_3 \cdot \bar{x}_4 \cdot y_2 \cdot \bar{y}_3 \vee x_5 \cdot y_2;$$

$$\varphi_1^0 = \bar{x}_3 \cdot x_4 \cdot \bar{x}_6 \cdot y_2 \cdot \bar{y}_3 \vee x_9 \cdot \bar{y}_2 \vee x_8 \cdot y_3;$$

$$\varphi_2^1 = x_3 \cdot \bar{y}_1 \vee x_6 \cdot y_1 \vee x_7 \cdot y_1 \vee x_8 \cdot y_3;$$

$$\varphi_2^0 = x_4 \cdot \bar{y}_1 \cdot \bar{y}_3 \vee x_{10} \cdot \bar{y}_1;$$

$$\varphi_3^1 = x_1 \cdot \bar{x}_1 \cdot \bar{y}_1 \cdot \bar{y}_2 \vee x_2 \cdot \bar{y}_1 \cdot y_2;$$

$$\varphi_3^0 = \bar{x}_1 \cdot x_2 \cdot y_1 \cdot \bar{y}_2 \vee x_3 \cdot \bar{y}_1 \vee x_5 \cdot y_2 \vee x_6 \cdot y_1 \vee x_7 \cdot y_1 \vee x_9 \cdot \bar{y}_2 \vee x_{10} \cdot \bar{y}_1,$$

де $\varphi_1^1, \varphi_2^1, \varphi_3^1$ і $\varphi_1^0, \varphi_2^0, \varphi_3^0$ — функції включення і виключення відповідно першого, другого і третього елементів пам'яті структурного автомата;

y_1, y_2, y_3 і $\bar{y}_1, \bar{y}_2, \bar{y}_3$ — сигнали на виходах першого, другого і третього елементів пам'яті, які відповідають логічним сигналам «1» і «0» відповідно;

z_1, z_2 — сигнали управління пристроєм.

Функція ϕ_1 відповідає елементу кода розміщеного зліва, а ϕ_3 — справа.

Рівняння включення і виключення елементів пам'яті отримують згідно із принципом, описаним у § 6.7, приклад 6.7.1.

Як видно із отриманих рівнянь, їх мінімізація не потрібна, тому переходимо до шостого кроку, де необхідно вибрати відповідну ПЛМ.

Виходячи з канонічних рівнянь роботи, ПЛМ повинна відповідати таким даним. Кількість кон'юнкторів у ній повинна бути не менше 26, диз'юнкторів — не менше 8, вхідних змінних — не менше 13. Таким властивостям відповідає ПЛМ, мікросхема серії K556PT1, яка має входи для 16 змінних, 8 виходів для реалізації восьми функцій і 48 кон'юнкторів.

Згідно з отриманими функціями $z_1, z_2, \phi_1^1, \phi_1^0, \phi_2^1, \phi_2^0, \phi_3^1, \phi_3^0$, присвоюємо номера їх кон'юнкторам:

$$\begin{aligned} k_1 &= \bar{y}_1 \cdot \bar{y}_2 \cdot \bar{y}_3; k_2 = \bar{y}_1 \cdot y_2 \cdot \bar{y}_3; k_3 = y_1 \cdot \bar{y}_2 \cdot y_3; k_4 = \bar{y}_1 \cdot \bar{y}_2 \cdot y_3; \\ k_5 &= \bar{y}_1 \cdot y_2 \cdot y_3; k_6 = y_1 \cdot y_2 \cdot \bar{y}_3; k_7 = \bar{x}_1 \cdot x_3 \cdot \bar{x}_4 \cdot y_2 \cdot \bar{y}_3; k_8 = x_5 \cdot y_2; \\ k_9 &= \bar{x}_3 \cdot x_4 \cdot \bar{x}_6 \cdot y_2 \cdot \bar{y}_3; k_{10} = x_9 \cdot \bar{y}_2; k_{11} = x_3 \cdot \bar{y}_3; k_{12} = x_6 \cdot y_1; k_{13} = x_7 \cdot y_1; \\ k_{14} &= x_8 \cdot y_3; k_{15} = x_4 \cdot \bar{y}_1 \cdot \bar{y}_3; k_{16} = x_{10} \cdot \bar{y}_1; k_{17} = x_1 \cdot \bar{x}_2 \cdot \bar{y}_1 \cdot \bar{y}_2; \\ k_{18} &= x_2 \cdot \bar{y}_1 \cdot y_2; k_{19} = \bar{x}_1 \cdot x_2 \cdot \bar{y}_1 \cdot \bar{y}_2; k_{20} = x_3 \cdot \bar{y}_1; k_{21} = x_5 \cdot y_2; k_{22} = x_6 \cdot y_1; \\ k_{23} &= x_7 \cdot y_1; k_{24} = x_9 \cdot \bar{y}_2; k_{25} = x_8 \cdot y_3; k_{26} = x_{10} \cdot y_1. \end{aligned}$$

Використовуючи рекомендації § 4.3 і дані § 6.8, програмуємо отримані функції і їх результати заносимо в табл. 6.8.2.

На невикористаних входах A14, ..., A16 з номерами кон'юнкторів $k_1, \dots, k_{26}$ перемички перепалюються.

На сьомому кроці будуюмо схему з використанням ПЛМ, мікросхема серії K556PT1. При побудові схеми використовуємо §4.4, § 4.5.

Таблиця 6.8.2

Кон'юнктори																Рівень активності											
Вхідна змінна																Вихідні функції											
Номер програмованого входу																Вихідні функції											
k_i	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	y_1	y_2	y_3	0	0	1	1	1	1	1	1	1	ϕ_1^0	ϕ_1^1	ϕ_2^0	ϕ_3^1	ϕ_3^0
	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	B1	B2	B3	B4	B5	B6	B7	B8	ϕ_3^0	ϕ_3^1	ϕ_3^0	ϕ_3^1	ϕ_3^0	ϕ_3^1
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	*	*	*	*	*	*
k_1	*	*	*	*	*	*	*	*	*	*	0	0	0	A	*	*	*	*	*	*	*	*	*	*	*	*	*
k_2	*	*	*	*	*	*	*	*	*	*	0	1	0	A	*	*	*	*	*	*	*	*	*	*	*	*	*
k_3	*	*	*	*	*	*	*	*	*	*	1	0	1	A	*	*	*	*	*	*	*	*	*	*	*	*	*
k_4	*	*	*	*	*	*	*	*	*	*	0	0	1	*	A	*	*	*	*	*	*	*	*	*	*	*	*
k_5	*	*	*	*	*	*	*	*	*	*	0	1	1	*	A	*	*	*	*	*	*	*	*	*	*	*	*
k_6	*	*	*	*	*	*	*	*	*	*	1	1	1	*	A	*	*	*	*	*	*	*	*	*	*	*	*
k_7	0	*	1	0	*	*	*	*	*	*	*	*	*	*	*	A	*	*	*	*	*	*	*	*	*	*	*
k_8	*	*	*	*	1	*	*	*	*	*	*	*	1	*	*	A	*	*	*	*	*	*	*	*	*	*	*
k_9	*	*	0	1	*	0	*	*	*	*	*	*	1	0	*	*	A	*	*	*	*	*	*	*	*	*	*
k_{10}	*	*	*	*	*	*	*	*	1	*	*	*	0	*	*	*	A	*	*	*	*	*	*	*	*	*	*
k_{11}	*	*	1	*	*	*	*	*	*	*	0	*	*	*	*	*	*	A	*	*	*	*	*	*	*	*	*
k_{12}	*	*	*	*	*	1	*	*	*	*	*	1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
k_{11}	*	*	*	*	*	*	1	*	*	*	1	*	*	*	*	*	*	*	*	*	*
k_{14}	*	*	*	*	*	*	*	1	*	*	*	*	1	*	*	*	*	*	*	*	*
k_{15}	*	*	*	*	0	*	*	*	*	*	0	*	0	*	*	*	*	*	A	*	*
k_{16}	*	*	*	*	*	*	*	*	*	1	0	*	*	*	*	*	*	*	A	*	*
k_{17}	1	0	*	*	*	*	*	*	*	*	0	0	*	*	*	*	*	*	*	A	*
k_{18}	*	1	*	*	*	*	*	*	*	*	0	1	*	*	*	*	*	*	*	A	*
k_{19}	0	1	*	*	*	*	*	*	*	*	0	0	*	*	*	*	*	*	*	*	A
k_{20}	*	*	*	1	*	*	*	*	*	*	0	*	*	*	*	*	*	*	*	*	A
k_{21}	*	*	*	*	*	*	*	*	*	*	1	*	*	*	*	*	*	*	*	*	A
k_{22}	*	*	*	*	*	1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	A
k_{23}	*	*	*	*	*	*	1	*	*	*	*	*	*	*	*	*	*	*	*	*	A
k_{24}	*	*	*	*	*	*	*	*	*	*	*	0	*	*	*	*	*	*	*	*	A
k_{25}	*	*	*	*	*	*	*	*	1	*	*	*	*	*	*	*	*	*	*	*	*
k_{26}	*	*	*	*	*	*	*	*	*	1	0	*	*	*	*	*	*	*	*	*	A

Наведена схема має вигляд, рис. 6.8.2.

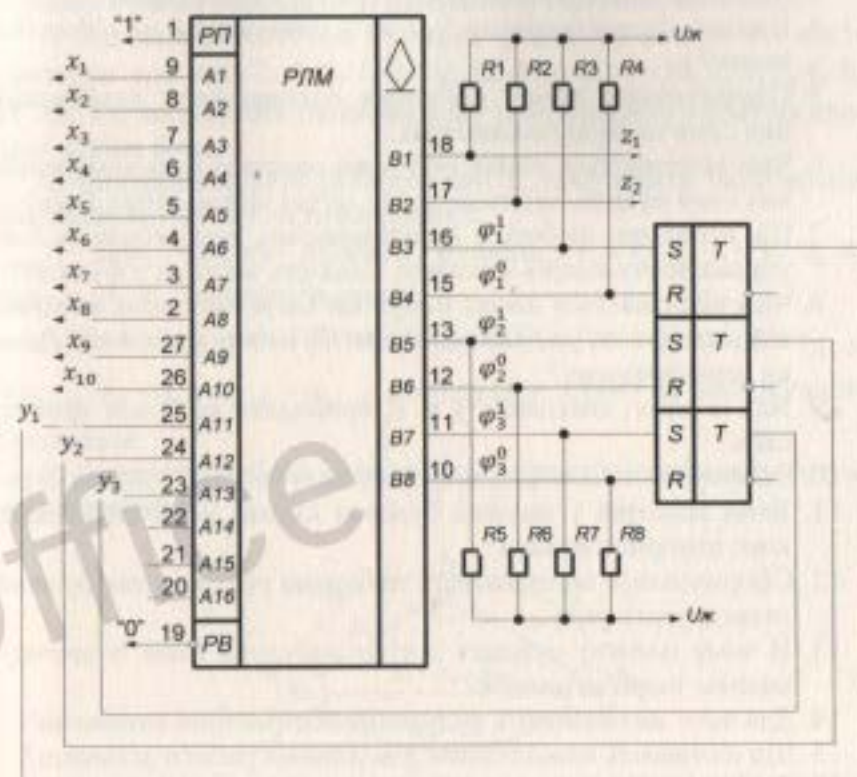


Рис. 6.8.2

де $R1, \dots, R8$ — резистори величиною 2 кОм; $U_{ж}$ — джерело живлення мікросхеми.

Із рис. 6.8.3 випливає, що схема реалізації пристрою із застосуванням ПЛМ виглядає значно простішою, ніж із застосуванням звичайних елементів логіки Буля. Такі схеми доцільно використовувати при значній кількості канонічних рівнянь.



Контрольні запитання

1. Сформулюйте логіку побудови одновихідних комбінаційних схем на елементах логіки Буля.

2. Назвіть базиси побудови одновихідних комбінаційних схем.
3. Чим відрізняється базис Шеффера від базиса Пірса?
4. В якому базисі доцільно будувати одновихідні комбінаційні схеми?
5. Сформулюйте логіку побудови одновихідних комбінаційних схем на мультиплексорах.
6. Чим відрізняється логіка побудови одновихідних комбінаційних схем на мультиплексорах від логіки на елементах Буля?
7. Що необхідно зробити в мультиплексорі, якщо кількість його управляючих входів менша за кількість змінних у функції?
8. Чим відрізняється логіка побудови багатовихідних комбінаційних схем на елементах логіки Буля від логіки побудови на дешифраторах?
9. Яка із логік, наведених у п. 8, приводить до більш простих схем?
10. Сформулюйте послідовність побудови часових булевих схем.
11. Який пристрій у часових булевих схемах необхідно додатково використовувати?
12. Сформулюйте послідовність побудови рекурентних булевих схем другого роду.
13. В чому полягає сутність логіки побудови схем із застосуванням теорії автоматів?
14. Для чого застосовують кодування абстрактних автоматів?
15. Що називають канонічними рівняннями роботи автомата?
16. Де і в яких випадках доцільно застосовувати програмовані логічні матриці при побудові комп'ютерних схем?



Задачі для самостійного розв'язування

1. Для логічної функції $y = x_1 \cdot \bar{x}_2 \cdot x_3 \vee \bar{x}_1 \cdot x_2 \vee x_1 \cdot \bar{x}_3$ побудувати одновихідну комбінаційну схему на елементах логіки Буля.
2. Використовуючи умову задачі 1, побудувати одновихідну комбінаційну схему в базисі Шеффера і Пірса.
3. Використовуючи умову задачі 1, побудувати одновихідну комбінаційну схему на мультиплексорі.
4. Для логічної функції, заданої в десятковій системі числення $y = \vee (0, 1, 3, 4, 5, 7)$, побудувати одновихідну комбінаційну схему

на елементах логіки Буля, у базисі Шеффера і на мультиплексорі. Отримані схеми порівняти за складністю і зробити висновки.

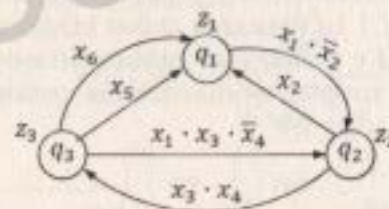
5. Для заданої системи логічних функцій у десятковій системі числення $y_1 = \vee (0, 2, 4, 5, 7, 9, 10, 12, 13)$; $y_2 = \vee (0, 1, 3, 5, 6, 8, 9, 11, 12, 14)$ побудувати багатовихідну комбінаційну схему на елементах логіки Буля.

6. Використовуючи умову задачі 5, побудувати багатовихідну комбінаційну схему на дешифраторі.

7. Для часової булевої функції $y_i = \bar{x}_1 \cdot \bar{x}_2 \cdot t_1 \vee x_1 \cdot \bar{x}_3 \cdot t_2 \vee \vee x_1 \cdot x_2 \cdot x_3 \cdot t_3$ побудувати схему.

8. Для рекурентної булевої функції другого роду $y_i = x_{2(i-2)} \vee \vee x_{2(i-1)} \cdot \bar{x}_{3i} \cdot x_{3i} \vee x_{3(i-1)} \vee x_{3(i-1)}$ побудувати схему із застосуванням D-тригерів.

9. Алгоритм роботи електронного пристрою заданий абстрактним автоматом.



Необхідно побудувати схему з використанням елементів логіки Буля і програмованої логічної матриці, мікросхеми серії K556PT1. Отримані схеми порівняти за складністю, зробити висновки.



Коментарі. Основні відомості, які викладені в цьому розділі з логіки побудови одновихідних і багатовихідних комбінаційних схем, взяті з [6, 16, 17], логіка побудови часових рекурентних булевих схем впливає з [17, 20], а логіка побудови схем із застосуванням теорії автоматів, в тому числі із застосуванням програмованих логічних матриць, — з [4, 12, 17, 21].

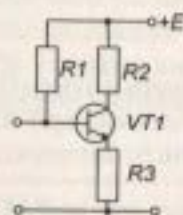


Розділ 7

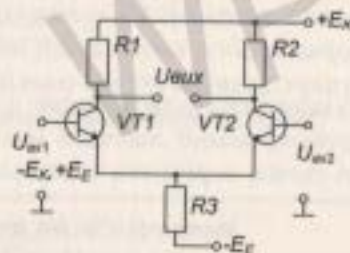
СХЕМОТЕХНІКА ПОБУДОВИ АНАЛОГОВИХ ВУЗЛІВ

7.1. Елементи схемотехніки аналогових вузлів

Основними елементами аналогових інтегральних мікросхем є транзистори та резистори. Характерною особливістю інтегральної аналогової схемотехніки є наявність взаємних компонентів. Під час виготовлення інтегральних мікросхем характеристики їх елементів досить близькі та взаємозалежні у разі зміни зовнішніх умов. Так, наприклад, на рис. 7.1.1а наведена схема підсилювального каскаду, де резистори $R1$ і $R2$ є взаємними компонентами, тому у разі зміни зовнішніх факторів коефіцієнт підсилення схеми практично не змінюватиметься, $K_u = -R2/R3$.



а)



б)

Рис. 7.1.1

Типовий диференційний каскад інтегральної мікросхеми, де взаємними компонентами, крім резисторів, є ще й транзистори $VT1$ і $VT2$, подано на рис. 7.1.1б. Маючи близькі та ідентичні характеристики, вони істотно зменшують температурний дрейф, дрейф від нестабільності джерел живлення та спрощують балансування каскаду.

Дрейф підсилювача здебільшого визначається вхідним каскадом — температурним дрейфом та часовою нестабільністю елементів. Дрейф диференційного каскаду на порядок менший від одиноконного. Але, якщо виготовити диференційний підсилювач в інтегральному виконанні, то його дрейф буде ще на порядок-два менший, ніж у диференційному виконанні з дискретними транзисторами.

На вході операційного підсилювача, як правило, встановлюють диференційний підсилювач, щоб ослабити наведення і підсилити U_{ex} . Типова схема диференційного вхідного каскаду в інтегральному виконанні наведена на рис. 7.1.2.

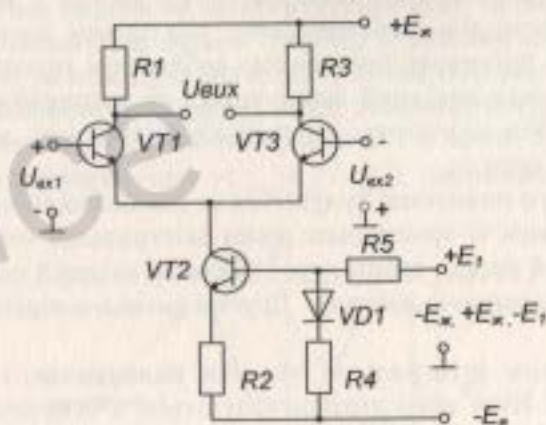


Рис. 7.1.2

Транзистор $VT2$ працює в режимі джерела струму, має великий внутрішній опір та поліпшує характеристики каскаду. Застосування елементів термокомпенсації ($VD1$, $R4$ і $R5$) дає змогу істотно поліпшити температурні характеристики всього каскаду.

7.2. Основні визначення та характеристики операційних підсилювачів

Операційні підсилювачі (ОП) постійного струму є досить поширеним базовим елементом аналогової схемотехніки, на якому можуть бути створені лінійні та нелінійні функціональні перетворювачі.

Означення 7.2.1. Операційним підсилювачем називають підсилювач постійного струму (ППС), який має високий коефіцієнт підсилення, два входи (так званий диференціальний вхід) і один вихід.

Зазвичай, ОП будують як ППС із безпосередніми зв'язками між каскадами, з диференціальним входом і біполярним відносно амплітуди підсилюваного сигналу виходом. Це забезпечує нульові потенціали на вході і виході ОП за відсутності вхідного сигналу. Тому такі підсилювачі легко з'єднувати послідовно, а також охоплювати зворотними зв'язками.

За своєю структурою ОП бувають три- або двокаскадні. За трикаскадною схемою будувались ОП в інтегральному виконанні першого покоління. Перший диференціальний каскад у них працює в режимі мікрострумів, забезпечуючи тим самим високий вхідний опір. Другий диференціальний каскад забезпечує підсилення напруги. Третій каскад вихідний, виконується як двотактний з спільним колектором. Він забезпечує підсилення потужності, а також низький вихідний опір.

ОП другого покоління будуються за двокаскадною схемою. Це стало можливим зі зростанням рівня інтегральної технології. При цьому перший каскад забезпечує і високий вхідний опір, і великий коефіцієнт підсилення напруги. Другий каскад є підсилювачем потужності.

З розвитком інтегральної техніки використання ОП значно розширилось. Нині вони використовуються, в основному, як високоякісні підсилювачі напруги в конструюванні будь-яких електронних пристроїв.

Поширеному застосуванню ОП сприяють їх високі параметри. Це великий коефіцієнт підсилення за напругою, що становить $K_v = (10^4 \dots 10^6)$; високий вхідний опір й кожному із входів $R_{вх} > 400 \text{ кОм}$; низький вихідний опір $R_{вих} < 100 \text{ Ом}$; досить широкий частотний діапазон — від нуля до одиниці мегагерц. За цими показниками ОП для багатьох застосувань наближаються до ідеального підсилювача, який має:

- 1) $K_v \rightarrow \infty$;
- 2) Два симетричних входи з $R_{вх} \rightarrow \infty$;
- 3) $R_{вих} \rightarrow 0$;
- 4) Безкінечний діапазон підсилюваного сигналу.

Функціональна схема такого підсилювача наведена на рис. 7.2.1.

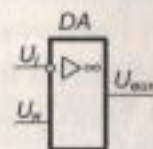


Рис. 7.2.1

Вхід, на який подано U_i , називають інвертуючим, а U_n — неінвертуючим.

Якщо сигнал подати на неінвертуючий вхід, то зміни вихідного сигналу співпадають за знаком (фазою) зі змінами вхідного. Якщо сигнал подати на інвертуючий вхід, то зміни вихідного сигналу матимуть протилежний знак (фазу) щодо зміни вхідного.

Найважливішими характеристиками ОП є вхідні амплітудні (передавальні) характеристики $-U_{вих} = f(U_{вх})$, зображені на рис. 7.2.2.

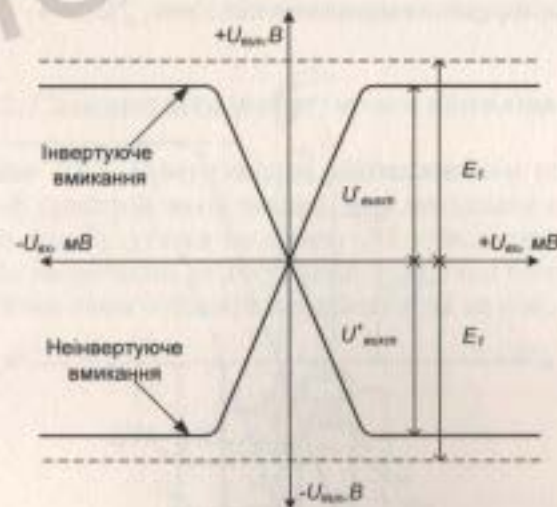


Рис. 7.2.2

Знімають ці характеристики, подаючи сигнали на один із входів і з'єднуючи інший з нульовою точкою.

Кожна вхідна характеристика має горизонтальну та скісну ділянки, які відповідають режиму повністю відкритого чи закритого

транзистора вихідного каскаду (режим насичення). Зі зміною напруги вхідного сигналу на цих ділянках вихідна напруга підсилювача залишається незмінною і визначається напругами $U_{\text{вих}}^+$ або $U_{\text{вих}}^-$, близькими до напруги джерела живлення E_1, E_2 .

Коефіцієнт підсилення в таких підсилювачах визначається за скісними ділянками і дорівнює:

$$K_s = \frac{\Delta U_{\text{вих}}}{\Delta U_{\text{вх}}}$$

Стан, за якого $U_{\text{вих}} = 0$ при $U_{\text{вх}} = 0$, називають балансом ОП. Однак, для реальних ОП умови балансу не виконуються (є розбаланс).

Напруга $U_{\text{вх}0}$, за якої $U_{\text{вих}} = 0$, має назву вхідної напруги зміщення нуля. Вона визначає напругу, яку необхідно подати на вхід підсилювача для створення балансу. Передавальні характеристики ОП за наявності розбалансу наведені на рис. 7.2.3.

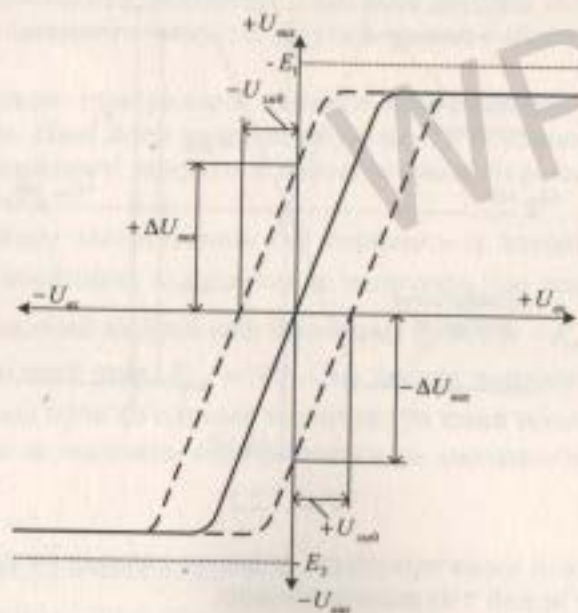


Рис. 7.2.3

$$U_{\text{вх}0} = \frac{\Delta U_{\text{вих}}}{K_s}$$

Корекція розбалансу відбувається ланцюгами, або за відсутності таких у ОП деяких типів, шляхом подачі на його вхід напруги, що дорівнює $U_{\text{вх}0}$ і протилежної за знаком. Вхідний опір, вхідний струм зміщення, максимальні вхідні, диференційна та синфазна напруги є основними вхідними параметрами ОП. За необхідності захисту від перенапруг між входами ОП вмикають зустрічно-паралельно включені два діоди або стабілітрони. Вихідними параметрами ОП є вихідний опір, максимальна вихідна напруга та струм.

Широке практичне використання ОП в аналогових схемах зумовлене, головним чином, застосуванням у їх схемах різного роду від'ємних зовнішніх зв'язків, чому сприяє велике значення коефіцієнта підсилення K_s , високий вхідний та малий вихідний опори. Висока якість параметрів сучасних ОП дозволяє, зокрема, без внесення помітної похибки при розрахунку схем на ОП, приймати $K_s \rightarrow \infty$, $R_{\text{вх}} \rightarrow \infty$, $R_{\text{вих}} \rightarrow 0$, а отже вважати опір за ідеальний.

7.3. Схемотехніка інвертуючого підсилювача

Означення 7.3.1. Інвертуючим підсилювачем називають схемотехнічний пристрій, який змінює знак вихідного сигналу відносно вхідного. Він будується на основі ОП з введенням паралельного від'ємного зворотного зв'язку, який подається безпосередньо на його вхід. Схема такого підсилювача наведена на рис. 7.3.1.

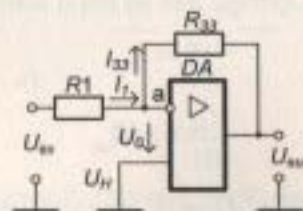


Рис. 7.3.1

Неінвертуючий вхід з'єднується зі спільною точкою схеми (точкою з нульовим потенціалом). Вхідний сигнал через резистор R

подається на інвертуючий вхід ОП. Кола живлення і ланцюги корекції тут і надалі не показано.

Виходячи з наведеного вище, а саме: вважаючи ОП за ідеальний, при аналізі схем з ОП слід виходити з таких положень:

- 1) коефіцієнт підсилення ОП нескінченний;
- 2) входи ОП струму не споживають ($R_{вх} = \infty$);
- 3) у вихідних колах ОП падіння напруги відсутнє ($R_{вих} = 0$);
- 4) якщо ОП охоплено від'ємним зворотним зв'язком, він працює в режимі підсилення, а не насичення, а різниця потенціалів між його входами $U_{вх} = U_0 = 0$.

Доведемо останнє положення.

$$U_{вх} = -K_u U_{вх};$$

$$U_{вх} = \frac{-U_{вих}}{K_u}$$

Якщо $K_u \rightarrow \infty$, то $U_{вх} \rightarrow 0$.

Реально $U_{вх} = U_0$ нулю не дорівнює. Але це настільки незначна величина, що для більшості схем на ОП нею можна знехтувати. Дійсно, якщо $U_{вих} = 10\text{В}$ (це майже відповідає насиченню), а $K_u = 100000$, то $U_0 = 100\text{ мкВ}$.

Оскільки на неінвертуючий вхід подана напруга $U_{нх} = 0$ (він з'єднаний із нульовою точкою), а $U_0 = 0$, то і потенціал інвертуючого входу також дорівнює нулю (віртуальний нуль). У результаті джерело входного сигналу пристроєм сприймається як $R1$ — вихідний опір підсилювача дорівнює величині резистора $R1$.

З першого закону Кіргофа для вузла a маємо:

$$I_1 = I_{33},$$

тобто

$$\frac{U_{вх}}{R1} = -U_{вих}.$$

ОП, забезпечуючи рівність $U_0 = 0$, створює на виході таку напругу, щоб відвести струм I_1 через резистор від'ємного зворотного зв'язку R_{33} . Тоді $K_u = -\frac{U_{вих}}{U_{вх}} = \frac{R_{33}}{R1}$.

Отже, K_u залежить лише від співвідношення опорів резисторів ділянки від'ємного зворотного зв'язку. Знак $\langle\langle - \rangle\rangle$ вказує на інверсію входного сигналу.

Вхідний опір схеми дорівнює величині $R1$.

Якщо $R > R1$, то $U_{вих} = \frac{-R_{33}}{R1} U_{вх} = U_{вх}$ — маємо інвертуючий масштабний підсилювач (із масштабним коефіцієнтом $K_u = -R1$).

При $R1$, $K_u = -1$ — схема набуває властивостей інвертуючого повторювача входної напруги (інвертора сигналу).

7.4. Схемотехніка неінвертуючого підсилювача

Означення 7.4.1. Неінвертуючим підсилювачем називають схемо-технічний пристрій, який не змінює знак вихідного сигналу відносно входного.

Він будується на основі ОП з введенням послідовного від'ємного зв'язку, який подають на інвертуючий вхід, а вихідний сигнал — на неінвертуючий вхід ОП. Схема такого підсилювача наведена на рис. 7.4.1а.

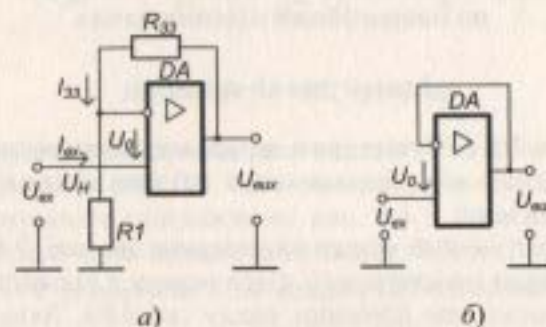


Рис. 7.4.1

Тут $U_{нх} = U_{вх}$, а вхідний струм $I_{вх} = 0$, бо $R_{вх} = \infty$.

Оскільки $U_0 = 0$, то $U_{R1} = U_{вх}$, а $U_{вх} / R1 = I_{33}$.

З іншого боку

$$I_{33} = \frac{U_{вих}}{R1 + R_{33}}.$$

Отже,

$$\frac{U_{ax}}{R1} = \frac{U_{max}}{R1 + R_{33}}.$$

Звідси

$$U_{max} = U_{ax} \left(1 + \frac{R_{33}}{R1} \right).$$

Тоді коефіцієнт підсилення неінвертуючого підсилювача

$$K_n = \frac{U_{max}}{U_{ax}} = 1 + R1.$$

Якщо $R1 \rightarrow \infty$, то отримаємо неінвертуючий повторювач, схема якого наведена на рис. 7.4.1б.

Неінвертуючий та інвертуючий підсилювачі широко використовуються як високостабільні підсилювачі різного призначення. Причому, неінвертуючий має великий вхідний опір (теоретично — нескінченний) і використовується для підсилення сигналів джерел із високим вихідним опором.

7.5. Схемотехніка побудови суматорів на операційних підсилювачах

Інвертуючий суматор

Означення 7.5.1. Суматором називають електронний пристрій, який призначений для підсилювання вхідних сигналів, які подаються на його входи.

Схема інвертуючого суматора наведена на рис. 7.5.1а. Він виконаний за типом інвертуючого підсилювача з кількістю паралельних гілок на вході, що дорівнює числу сигналів. Якщо опори всіх резисторів схеми однакові, то

$$R_{33} = R1 = R2 = \dots = R_n \ll R_{ex}.$$

то при $I_{ax} = 0$ маємо

$$I_{33} = I_1 + I_2 + \dots + I_n$$

або

$$U_{max} = -(U_1 + U_2 + \dots + U_n).$$

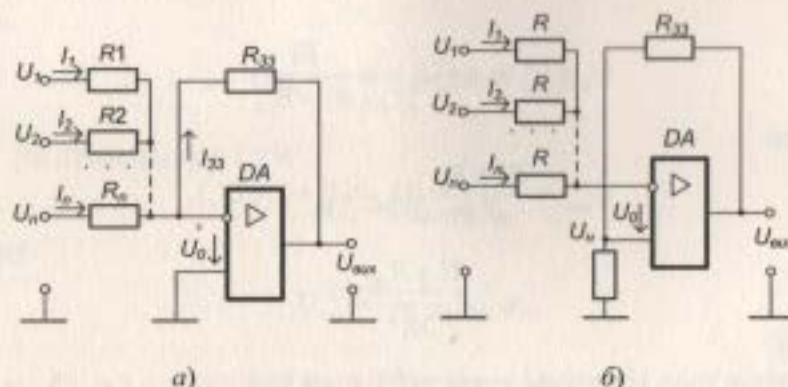


Рис. 7.5.1

Останнє співвідношення показує рівноправну вагову участь доданків у їх сумі. Підсумовування може виконуватись також із різними ваговими коефіцієнтами для кожного з доданків.

Досягається це використанням різних значень опорів резисторів у вхідних гілках

$$U_{max} = - \left(\frac{R_{33}}{R1} U_1 + \frac{R_{33}}{R2} U_2 + \dots + \frac{R_{33}}{R_n} U_n \right).$$

Неінвертуючий суматор

Неінвертуючий суматор можна отримати шляхом послідовного з'єднання суматора рис. 7.5.1 а та інвертора, рис. 7.3.1. Але на основі неінвертуючого підсилювача, рис. 7.4.1, його можна створити значно простіше — як це показано на рис. 7.5.1б.

При $U_0 = 0$ напруга на обох входах ОП однакова і становить

$$U_n = \frac{U_{max} \cdot R1}{R_{33} + R1}.$$

Оскільки струм неінвертуючого входу дорівнює нулю (тому що $R_{ex} \rightarrow \infty$), то маємо

$$\frac{U_1 - U_n}{R} + \frac{U_2 - U_n}{R} + \dots + \frac{U_n - U_n}{R} = 0$$

або

$$U_1 + U_2 + \dots + U_n = n \frac{R_1}{R_1 + R_{33}} U_{\text{вх}}.$$

звідси

$$U_{\text{вх}} = \frac{R_1 + R_{33}}{n \cdot R_1} (U_1 + U_2 + \dots + U_n),$$

$$\frac{R_1 + R_{33}}{n \cdot R_1} = 1$$

і тоді

$$U_{\text{вх}} = U_1 + U_2 + \dots + U_n.$$

7.6. Схемотехніка побудови інтеграторів, диференціаторів і компараторів на операційних підсилювачах

Інтегратор

Означення 7.6.1. Інтегратором називають електронний пристрій, призначений для інтегрування вхідного сигналу.

Схема інтегратора наведена на рис. 7.6.1. Вона створюється заміною в схемі інвертуючого підсилювача, рис. 7.3.1, резистора зворотного зв'язку R_{33} конденсатором C .

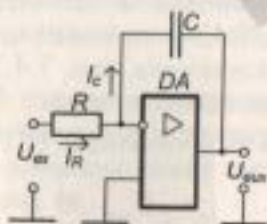


Рис. 7.6.1

Оскільки $R_{\text{вх}} = \infty$, то маємо

$$I_R = I_C \text{ і } \frac{U_{\text{вх}}}{R} = -C \frac{du_{\text{вих}}}{dt}$$

або

$$U_{\text{вих}} = -\frac{1}{RC} \int u_{\text{вх}} dt + U_{\text{вих}0}.$$

Як правило, при $t = 0$

$$U_c = U_{\text{вих}0} = 0$$

тому

$$U_{\text{вих}} = -\frac{1}{RC} \int u_{\text{вх}} dt,$$

де $RC = \tau$ — постійна часу. Реальному масштабу часу відповідає $\tau = 1$ с.

При подачі на вхід постійної напруги, струм, що заряджає конденсатор, має постійну величину $U_{\text{вх}} / R$ (не залежить від ступеня заряду конденсатора), і конденсатор заряджається рівномірно, а вихідна напруга зростає лінійно.

Тому інтегратор часто застосовують як основу генераторів лінійних напруг.

На рис. 7.6.2 зображені часові діаграми роботи інтегратора при подачі на його вхід постійної напруги.

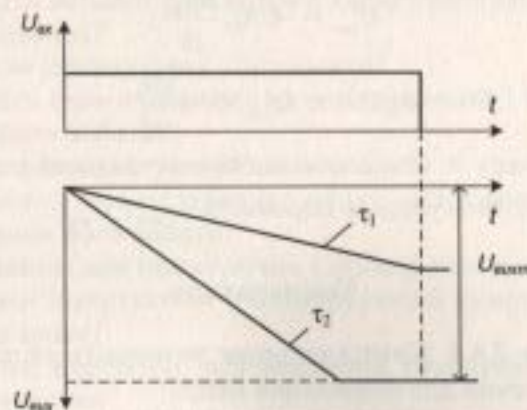


Рис. 7.6.2. $U_{\text{вих}}$

При τ_2 — параметри схеми вибрані неправильно, бо не забезпечується виконання інтегрування за весь час дії вхідного сигналу (ОП входить у режим насичення).

Диференціатор

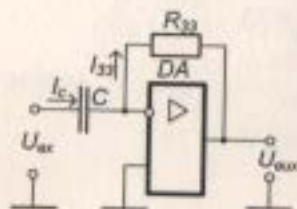


Рис. 7.6.3

Означення 7.6.2. Диференціатором називають електронний пристрій, призначений для диференціювання вхідного сигналу.

Схема диференціатора наведена на рис. 7.6.3. Від схеми інтегратора вона відрізняється заміною місцями резистора і конденсатора.

В цій схемі інтегратора

$$I_c = I_{33};$$

$$I_c = -C \frac{du_{\text{вх}}}{dt};$$

$$I_{33} = \frac{U_{\text{вх}}}{R_{33}};$$

$$-C \frac{du_{\text{вх}}}{dt} = \frac{U_{\text{вх}}}{R_{33}};$$

$$U_{\text{вих}} = -CR_{33} \frac{du_{\text{вх}}}{dt};$$

$$R_{33}C = \tau, \quad U_{\text{вих}} = -\tau \frac{du_{\text{вх}}}{dt}.$$

Постійну часу τ необхідно вибирати так, щоб у процесі диференціювання дотримувалась нерівність $U_{\text{вих}} < U_{\text{ном}}^-$.

Компаратори

Означення 7.6.3. Компаратором називають електронний пристрій, призначений для порівняння напруг.

Схема найпростішого компаратора зображена на рис. 7.6.4. Він виконує порівняння вхідного сигналу $U_{\text{вх}}$ з опорною напругою $U_{\text{оп}}$. Сигнал на виході ОП змінює полярність, коли ці напруги зрівнюються, як показано на часових діаграмах роботи компаратора, наведених на рис. 7.6.3б.

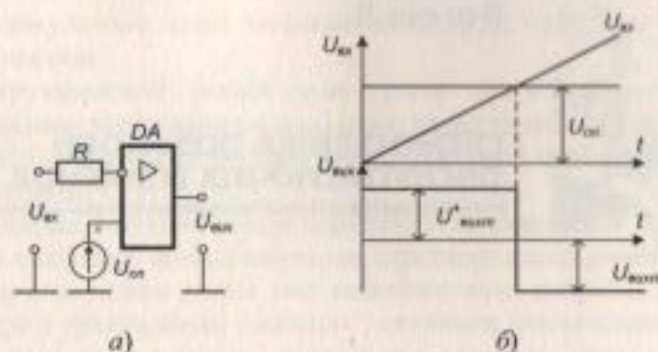


Рис. 7.6.4



Контрольні запитання

1. Що таке дрейф нуля підсилювача постійного струму?
2. Що таке диференційний каскад підсилювача постійного струму? Накресліть його схему і поясніть принцип її дії?
3. Що таке операційний підсилювач, як він побудований і які його властивості?
4. Наведіть основні параметри і характеристики операційного підсилювача?
5. Що таке інвертуючий підсилювач?
6. Наведіть схему інвертуючого підсилювача і поясніть принципи його роботи?
7. Що таке неінвертуючий підсилювач?
8. Накресліть схему неінвертуючого підсилювача і поясніть принципи його роботи?
9. Яка різниця між інвертуючим і неінвертуючим підсилювачем?
10. Що таке інвертуючий і неінвертуючий суматор? Яка різниця між ними?
11. Що таке інтегратор, диференціатор, компаратор? Де вони застосовуються?



Коментарі. Елементи схемотехніки аналогових вузлів взяті з [9, 15, 26], основні визначення та характеристики впливають із [15], а схемотехніка інвертуючого і неінвертуючого підсилювача, схемотехніка побудови суматорів, інтеграторів, диференціаторів та компараторів — із [15, 23].



Розділ 8

СХЕМОТЕХНІКА ПОБУДОВИ ОБСЛУГОВУЮЧИХ ЕЛЕМЕНТІВ

Значну частину в сучасних комп'ютерах становлять блоки управління, індикації та контролю. У цих блоках використовують схеми, які виконують різні спеціальні функції, наприклад, перетворення рівнів, генерування сигналів, формування різних керуючих сигналів, сигналів із параметрами. Вимоги до побудови обслуговуючих елементів дуже різноманітні, тому їх рівень інтеграції і номенклатура значно нижчі, ніж аналогічні параметри для логічних елементів.

Важливою умовою під час розроблення спеціальних обслуговуючих елементів є сумісність їх за входами і виходами з логічними елементами, мікропроцесорами, на основі яких проектують комп'ютерний пристрій. У зв'язку з цим особливу увагу приділяють реалізації обслуговуючих елементів на базі стандартних логічних елементів.

8.1. Схемотехніка перетворювачів рівнів

Означення 8.1.1. Перетворювачами рівнів називають спеціальні елементи цифрових пристроїв, які призначені для забезпечення сумісності логічних рівнів різних типів цифрових елементів.

Нині логічні рівні ТТЛ-елементів та їх навантажувальні характеристики є стандартними для цифрових пристроїв, мікропроцесорів, мікро-ЕОМ тощо, незалежно від їх технології і схемотехніки елементної бази.

Більшість інтегральних схем із високим рівнем інтеграції виконано на основі p, n або КМОН-технології, тоді як схеми малого і середнього рівнів інтеграції виконані на основі ТТЛ і КМОН-технології. Є також значне число схем інших типів, тому розглянути всі варіанти перетворювачів рівня неможливо. У зв'язку з

цим сформулюємо деякі загальні правила їх побудови для більшості випадків:

1. Перетворювачі рівнів проектують для конкретних схем з обов'язковим урахуванням вихідних характеристик і параметрів керуючого елемента та вхідних характеристик і параметрів керованого елемента;

2. Перепад логічних рівнів керуючого елемента має бути достатнім для надійного функціонування перетворювачів рівнів;

3. Перетворювач рівнів має забезпечувати потрібні динамічні параметри з урахуванням ємнісних і активних навантажень.

У складі схем малого і середнього ступеня інтеграції ТТЛ і КМОН-типу є спеціально розроблені перетворювачі рівнів. Так, наприклад, перетворювачі КМОН-ТТЛ це: 176ПУ1; 176ПУ2; 176ПУ3; 564ПУ4; 564ЛН2; перетворювачі ТТЛ-КМОН: 133ЛН3, 133ЛН5 та ін., фрагменти функціональних схем одних із яких наведені на рис. 8.1.1.

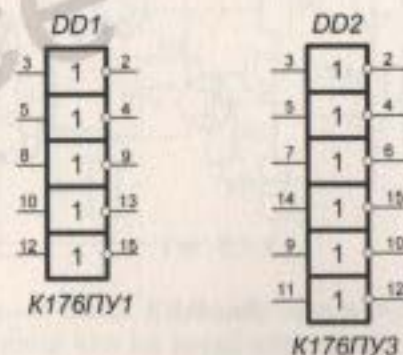


Рис. 8.1.1.

На рис. 8.1.2, наведено приклад стикування КМОН-схем, що працюють на високому рівні напруги із КМОН-схемами, що працюють з низьким рівнем напруги джерел живлення. Резистори $R1$ і $R2$ можна рекомендувати тут рівними $R1 = R2 = 20 \text{ кОм}$. На рис. 8.1.3 наведено схему перетворювача рівня КМОН-ТТЛ.

У цій схемі наведено приклад перетворення високого рівня логічних сигналів КМОН-структури з напругою живлення 15В — в сигнали логічного рівня ТТЛ-логіки з напругою 5В. Параметри резисторів цієї схеми мають такі значення: $R1 = 51 \text{ кОм}$; $R2 = 10 \text{ кОм}$; $R3 = 4 \text{ кОм}$; $R4 = 1,6 \text{ кОм}$; $R5 = 1 \text{ кОм}$; $R6 = 150 \text{ Ом}$.

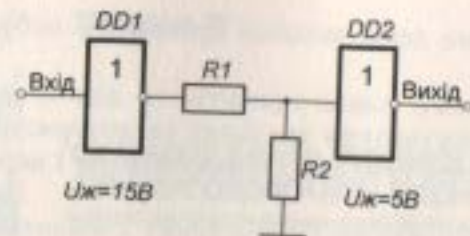


Рис. 8.1.2

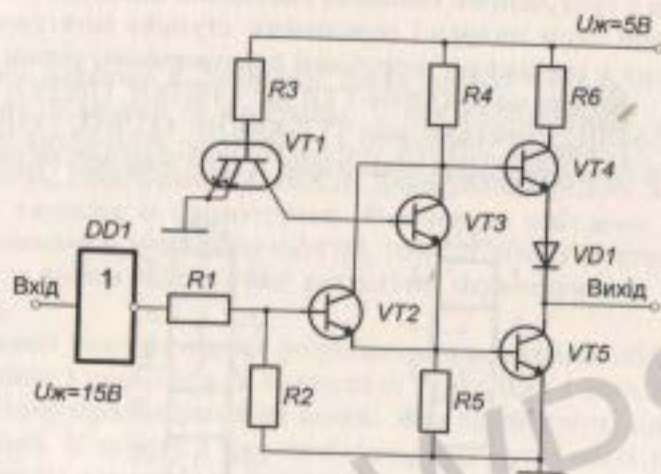


Рис. 8.1.3

Схема працює таким чином. При подачі логічного сигналу «0» на вхід логічного елемента DD1 КМОН структури на його виході з'являється логічний сигнал «1», що приводить до відкриття транзисторів VT2 і VT5, тобто на виході TTL-логіки формується сигнал логічного «0». При подачі сигналу «1» на вхід логічного елемента DD1 КМОН-структури на його виході з'являється сигнал «0», що приводить до закриття транзисторів VT2, VT5 і появи на виході TTL-логіки логічного сигналу «1».

Активним елементом перетворювача рівнів у розглянутій схемі є будь-яка ТТЛ-схема, яка має виходи розширення «АБО». Тоді інформаційні входи ТТЛ-схеми необхідно заземлити, у результаті чого транзистор VT_1 буде завжди закритий. Зовнішній транзистор VT_2 , наприклад КТ315А, приєднується до розширювальних входів

ТТЛ-схеми. Керування транзистором $VT2$ здійснює подільник $R1, R2$, підключений до виходу керуючої КМОП-схеми.

Якщо необхідно узгодити різні логічні схеми, напруги в яких різного знака, або в разі значного логічного перепаду в кожній зі схем, то для цих випадків використовують транзисторні перемикальні схеми, як перетворювачі рівнів. Одна зі схем таких перетворювачів рівнів, в яких за допомогою елементів ТТЛ-типу, наприклад, мікросхеми серії К155, управляють виконавчим пристроєм, що спрацьовує від напруги 27В з вхідним опором $R_1 = 1\text{кОм}$, наведена на рис. 8.1.4.

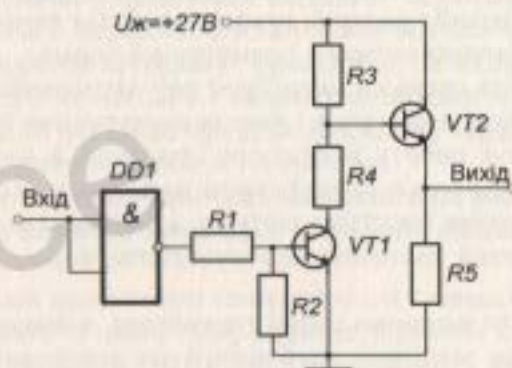
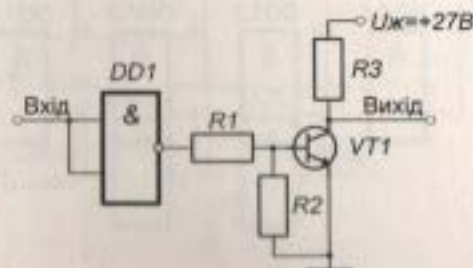


Рис. 8.1.4

Це перемикальна схема з транзисторами $n-p-n$ -типу працює таким чином. При рівні «1» на вході керуючого елемента ТТЛ-типу транзистори $VT1$ і $VT2$ закриті і на вході виконавчого пристрою буде логічний сигнал «0», так як колекторний струм транзистора $VT2$ практично дорівнює «0». При рівні «0» на вході керуючого елемента ТТЛ-типу транзистори $VT1$ і $VT2$ відкриті і насичені, а на резисторі $R5$ буде напруга близькою до $U_{ж} = 27$ В, що відповідає сигналу логічної «1».

Якщо непотрібно прив'язувати вихід виконавчого пристрою до землі, то перетворювач рівня можна виконати на одному транзисторі, рис. 8.1.5.



Proc. 8.1.5

Схема працює таким чином. При подачі логічної «1» на вхід керуючого елемента ТТЛ-типу, транзистор $VT1$ закритий і на його виході формується сигнал логічної «1», близький до 27В. Якщо на вхід елемента ТТЛ-типу подати сигнал логічного «0», то транзистор $VT1$ вмикється і на його колекторі формується сигнал логічного «0».

8.2. Схемотехніка генераторів сигналів прямокутної форми

Означення 8.2.1. Генератором сигналів прямокутної форми називають цифровий пристрій, призначений для формування послідовності електричних сигналів прямокутної форми.

Послідовність сигналів може бути регулярною або з перериванням, або зі зміною параметрів і форми електричних сигналів. Генератор забезпечує роботу цифрового пристрою в часі за законом, зумовленим внутрішнім структурним пристроєм. Кожний генератор характеризується частотою сигналу, стабільністю частоти, можливістю керування частотою, шпоруватістю, видом послідовності сигналів і т. п.

На рис. 8.2.1а наведено схему генератора, в якому конденсатор C забезпечує час затримки, необхідний для створення позитивного зворотного зв'язку, і від його ємності залежить частота генерації.

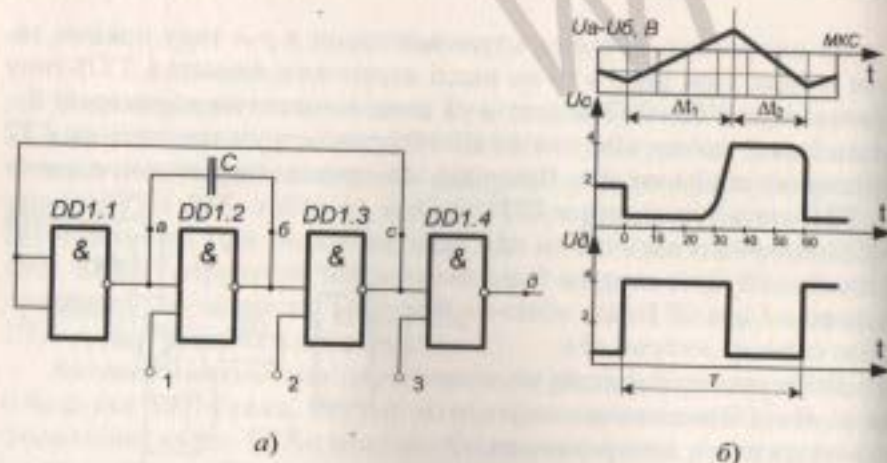


Рис. 8.2.1

Дана схема генератора виконана на мікросхемі K155ЛА3 при $C = 0,47$ мкФ. Сам генератор виконаний на трьох елементах «2І-НІ», DD1.1, ..., DD1.3. Четвертий елемент «2І-НІ» використовують для поліпшення форми вихідного сигналу. При роботі генератора на входи 1 і 2 потрібно подати логічний сигнал «1» (за подавання на ці входи логічного сигналу «0» генерація зривається і на виході d фіксується постійний логічний рівень. Якщо на вхід 2 подається логічний сигнал «0», то на виході d встановлюється логічний рівень «0» за наявності логічного сигналу «1» на виході 3 логічного елемента DD1.4.

Генератор може управлятися входом 3. За подачі логічного сигналу «0» на вхід 3 на виході d встановлюється логічний сигнал «1». В цей час генератор продовжує працювати. За подачі логічного сигналу 1 на вхід 3 на виході d з'являються сигнали генератора.

На рис. 8.2.1б наведено процеси розрядки і зарядки конденсатора C та часові діаграми роботи генератора.

Перевагою схеми генератора, наведеного на рис. 8.2.1, є її простота, тому що потрібен лише один зовнішній елемент — конденсатор C .

До недоліків даної схеми генератора слід віднести:

1. Шпаруватість цього генератора не дорівнює 2.
2. Логічні елементи DD1.1, DD1.2 працюють у критичному режимі, що є істотним недоліком, з якого забороняється використовувати цю схему в апаратурі, що працює у складних умовах експлуатації.
3. У жодній точці схеми, за винятком точки C , немає добре сформованого сигналу.

Усі зазначені недоліки можна усунути шляхом вмикання на вхід критичних логічних елементів DD1.1 і DD1.2, резисторів $R1$ і $R2$ (не обов'язково однакових). Схема такого генератора наведена на рис. 8.2.2.

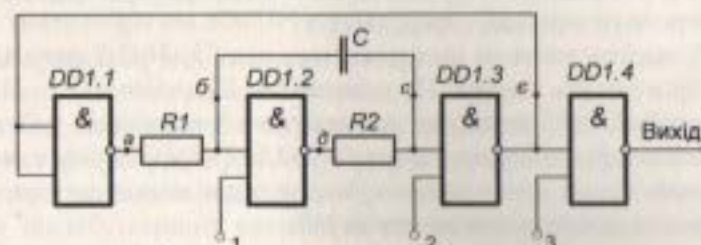


Рис. 8.2.2

Для забезпечення умов генерації схеми наведеної на рис. 8.2.2, резистори R слід вибрати такої величини, щоб для ізольованого елемента «2I-НІ» при заземленому резисторі на його вході забезпечувався рівень, менший ніж $U_{пор.}$ для найгірших умов експлуатації. Часова діаграма роботи схеми такого генератора наведена на рис. 8.2.3.

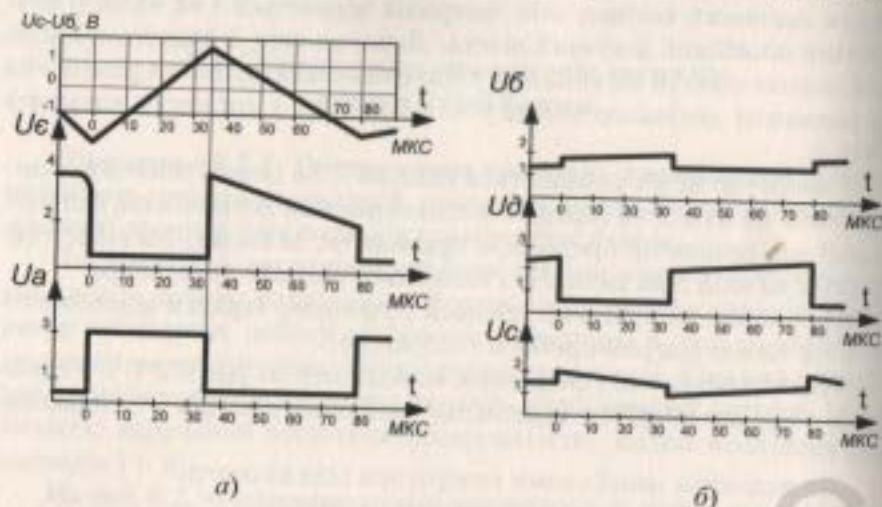


Рис. 8.2.3

Порівнюючи часові діаграми роботи генераторів на рис. 8.2.1б і 8.2.3б, можна зробити висновок, що підбором резисторів R неважко забезпечити шпаруватість їх роботи величини 2. Часові діаграми роботи генератора в полегшеному режимі відповідають таким елементам схеми $R1 = R2 = 300 \text{ Ом}$; $C = 0,47 \text{ мкФ}$; елементи DD1.1, ..., DD1.4 — мікросхемам серії K155ЛА3. При цих значення резисторів у схемі струми заряду і розряду конденсатора C знижені приблизно в п'ять разів, і тому схема такого генератора є високонадійною для роботи в різних середовищах.

Стабільність частоти вихідних коливань для розглянутих типів генераторів досить низька. Це пояснюється як значним технологічним розкидом, так і великою залежністю від зовнішніх дестабілізуювальних факторів, зокрема температури, параметрів усіх елементів, що задають час. З цих причин сумарне відхилення частоти від необхідного значення може досягати 10%.

Іноді для синхронізації роботи складних цифрових пристроїв потрібна дуже висока стабільність частоти генератора, наприклад,

0,001% і менше. На практиці існує багато способів стабілізації частоти вихідної напруги генераторів. Але найпростішим і найефективнішим з них є застосування кварцової стабілізації. Сутність цього способу полягає в тому, що в якості елемента, який задає час у генераторі, використовують кварцовий резонатор. Типові схеми таких генераторів наведені на рис. 8.2.4.

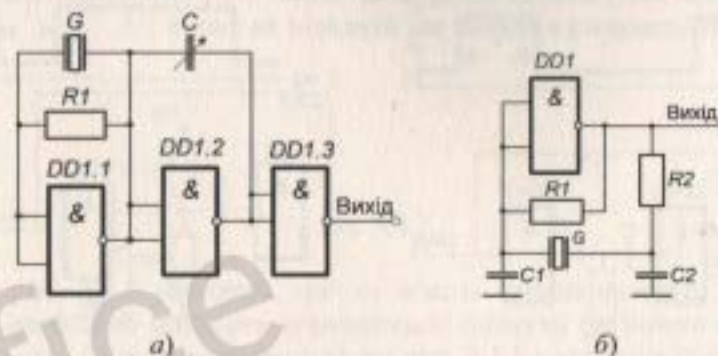


Рис. 8.2.4

Частота вихідної напруги в такому генераторі визначається параметрами кварцового резонатора G . Резонатор G вибирають з умови надійного виникнення коливань. Зміною ємності конденсатора C можна в незначній мірі підстроювати частоту вихідних коливань генератора. Логічний елемент DD1.3 ТТЛ-типу є буферним і призначений для поліпшення форми вихідних коливань генератора.

Як правило, для генераторів застосовуються малопотужні елементи ТТЛ-типу, що дає змогу отримувати вихідні імпульси з частотою не більше ніж 1, ..., 10 МГц.

Для отримання більш високої стабільності і частоти використовують генератори на КМОН-структурах. Одна із таких схем наведена на рис. 8.2.4б. Схема має такі дані за елементами: резистор $R1 = 10, \dots, 30 \text{ МОм}$; $R2 = 510 \text{ кОм}$; $C1 = 36, \dots, 62 \text{ пФ}$; $C2 = 5, \dots, 62 \text{ пФ}$; G — кварцовий генератор із частотою 32 768 Гц; логічний елемент серії K561ЛА7 (КМОН-структури). Такий варіант схеми забезпечує не тільки високу стабільність, але й виключає необхідність у підзабудованих елементах шляхом поділу на більш високу частоту генератора до необхідного значення за допомогою лічильників.

8.3. Схемотехніка одновібраторів

Означення 8.3.1. Одновібратором називають електронний пристрій, що виробляє вихідний імпульс за одиничним перепадом вхідного сигналу. Тривалість вихідного імпульсу визначається постійною часу $R \cdot C$ вбудованих або зовнішніх компонентів. Однією із можливих схем одновібраторів на основі логічних елементів ТТЛ-типу і RC -ланцюга є схеми, які наведені на рис. 8.3.1.

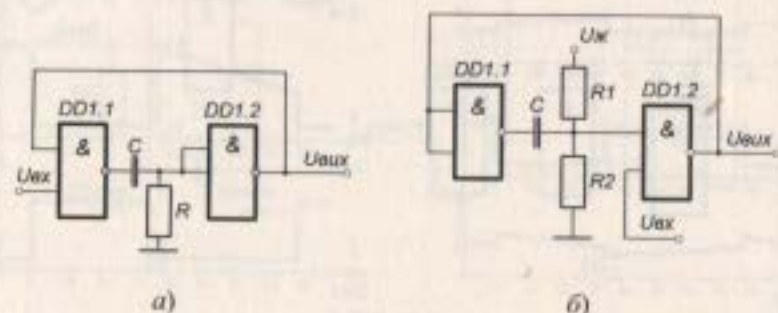


Рис. 8.3.1

Опір резистора R у схемі на рис. 8.3.1а вибирають таким, щоб в установленому режимі на виході генератора схеми $DD1.2$ був сигнал логічної «1». Але оскільки резистор R є навантажувальним для елемента $DD1.1$, то його опір не може бути меншим за величину, за якої рівень логічної «1» елемента $DD1.1$ знижується до допустимої мінімальної величини, наприклад, 2,4В для елементів ТТЛ-типу. Для серії мікросхем К155 опір даного резистора дорівнює $R = 1,2 \text{ кОм}$.

Опір резистора $R2$ в схемі одновібратора наведений на рис. 8.3.1б, на відміну від попередньої схеми повинен бути таким, щоб логічний елемент $DD1.2$ в установленому режимі мав на своєму виході сигнал логічного «0», тобто потенціал на виході резистора $R2$ повинен бути 2,4В плюс порогове значення логічного «0» елемента $DD1.2$. Виходячи з цих умов, резистори $R1$ і $R2$ повинні мати величини 2 кОм і 2,8 кОм відповідно для ТТЛ-типу мікросхем серії К155.

Часові діаграми роботи даних схем приведені на рис. 8.3.2.

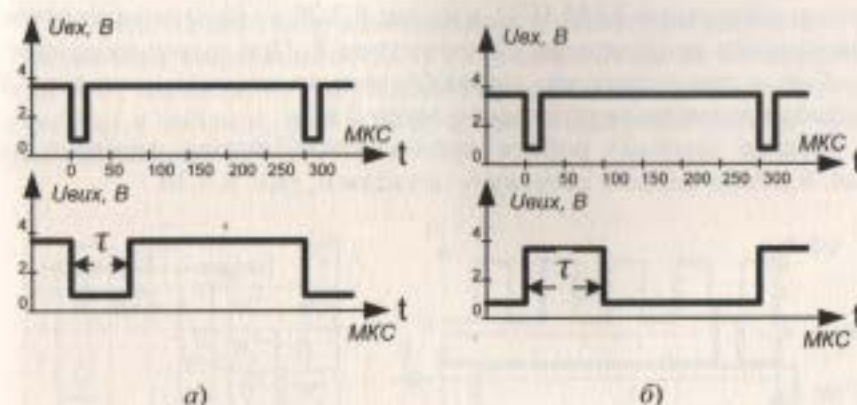


Рис. 8.3.2

Із рис. 8.3.2 виходить, що на обидві схеми, наведені на рис. 8.3.2, необхідно подавати короткочасні сигнали логічного «0» для їх запуску. Схема, яка наведена на рис. 8.3.1а видає на своєму виході сигнал затримки τ у вигляді логічного «0», рис. 8.3.2а, а схема рис. 8.3.1б — сигнал τ у вигляді логічної «1» рис. 8.3.2б.

У складі деяких серій сучасних інтегральних мікросхем є одно — вібратори двох типів: без повторного і з повторним запуском. Це мікросхеми К155АГ1 і К155АГ3 відповідно для мікросхем ТТЛ-типу. На рис. 8.3.3 показані варіанти ввімкнення одновібратора в роботу на базі мікросхеми К155АГ1.

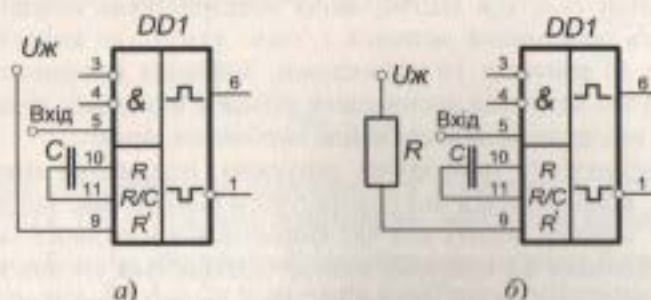


Рис. 8.3.3

На рис. 8.3.2а показаний варіант ввімкнення зовнішнього конденсатора C за внутрішнього резистора $R = 2 \text{ кОм}$ (внутрішній ре-

зистор мікросхеми K155АГ1), а на рис 8.3.2б — варіант ввімкнення зовнішнього конденсатора C і резистора R . При цьому ввімкненні необхідно при розрахунку одновібратора до значення резистора R додавати значення внутрішнього опору 2 кОм.

Часова діаграма роботи такого одновібратора наведена на рис. 8.3.3а, а варіант запуску — в таблиці, рис. 8.3.3б.

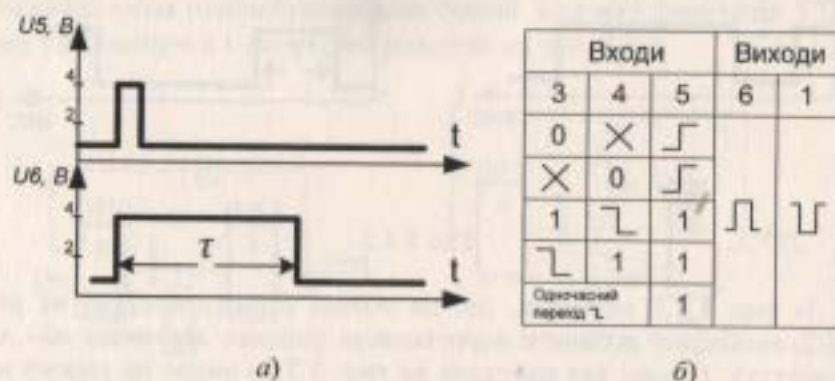


Рис. 8.3.3

Із таблиці, рис. 8.3.3б, випливає, що за будь-яких комбінацій статичних сигналів на входах 3, 4 і 5 одновібратор перебуває у стабільному стані, за якого на його виході 1 виходить сигнал логічного «0».

Згідно технічних умов на мікросхему $R_{вх} = 1,4$ кОм, а $R_{вх2} = 30$ кОм. Ємність конденсатора C має бути менше або дорівнювати 1000 мкФ, причому допускається застосування електролітичних конденсаторів. Полярність ввімкнення останніх є така: «+» — до контакту 11, а вивід «-» до контакту 10 мікросхеми. Значення конденсатору $C = 10, \dots, 1000$ мкФ слід застосувати тільки у випадках, коли до стабільності вихідних імпульсів немає особливих вимог.

Одновібратор з повторним запуском, наприклад мікросхема K155АГ3, відрізняються від розглянутого раніше тим, що реагує на перепади запуску, навіть під час формування вихідного імпульсу. У цьому випадку на прямому виході залишається сигнал високого рівня як завгодно довго, якщо час між перепадами запуску буде меншим, ніж тривалість вихідного сигналу, реалізованого від одиничного перепаду запуску, з урахуванням часу відновлення одновібратора. Іншою відмінністю є те, що цей одновібратор можна повернути у верхній стан у будь-який момент часу сигналом скидання.

Функціональна схема одновібратора з повторним запуском з застосуванням мікросхеми K155АГ3 і підключенням до неї компонентів R і C , наведена на рис. 8.3.4а. Варіанти запуску одновібратора наведені в таблиці, рис. 8.3.4з, а часові діаграми роботи — на рис. 8.3.4б і рис. 8.3.4в.

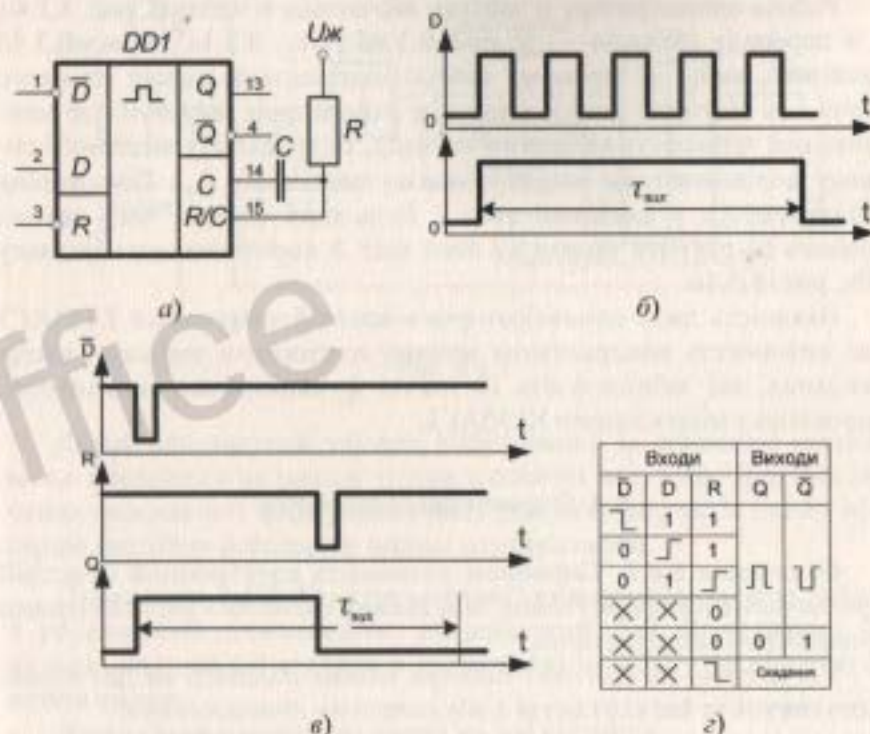


Рис. 8.3.4

Запуск одновібратора здійснюється негативним перепадом $\bar{D}$ при $D = «1»$, і $R = «1»$, або позитивним перепадом на вході D при $\bar{D} = «0»$, і $R = «1»$, або позитивним перепадом на вході R при $\bar{D} = «0»$ і $D = «1»$, рис. 8.3.4 з. За будь-яких комбінацій статичних сигналів на входах $\bar{D}$, D і R одновібратор перебуває у стабільному стані, за яким $Q = «0»$, $\bar{Q} = «1»$. Зовнішні компоненти R і C визначають тривалість вихідного імпульсу. У тому випадку, коли потрібно отримати імпульс великої тривалості, а до стабільності $\bar{D}$ не висувають

жорстких вимог, слід використовувати електролітичні конденсатори через малі габарити, але з напругою не нижче 1В. За цього позитивний вивід конденсатора з'єднують з контактом 15 одновібратора, а негативний — з контактом 14. Згідно технічних умов на мікросхему $R_{min} = 5 \text{ кОм}$, а $R_{max} = 25 \text{ кОм}$. Обмежень на величину ємності конденсатора C не накладається.

Робота одновібратора зі запуску вбачається із таблиці, рис. 8.3.4г, а зі перепаду сигналів — на рис. 8.3.4б і рис. 8.3.4в. Із рис. 8.3.4б виходить, якщо на прямому виході залишається сигнал високого рівня і як завгодно довго, а час між перепадами запуску буде меншим, ніж тривалість вихідного сигналу, то тривалість вихідного сигналу дорівнюватиме розрахунковому значенню, τ_{max} . Повернення одновібратора у вихідний стан у будь-який момент часу можна зробити за рахунок подачі на його вхід R короткочасного сигналу «0», рис. 8.3.4в.

Наявність двох одновібраторів в корпусі мікросхеми K155АГ3 дає можливість використання режиму повторного запуску і входу скидання, які забезпечують їй значні функціональні можливості, порівняно з мікросхемою K155АГ1.

8.4. Схемотехніка таймерів

Означення 8.4.1. Таймером називають електронний пристрій призначений для формування імпульсних сигналів з регульованими тривалістю і шпаруватістю.

Усі існуючі на сьогодні таймери можна поділити на два класи: **однотактні** та **багатотактні** з вбудованими лічильниками.

Означення 8.4.2. Однотактним таймером називають таймер, який призначений для формування часових інтервалів тривалістю від одиниць мікросекунди до одиниць години.

Однотактні таймери являють собою комбінацію аналогової частини (компаратора) з цифровою послідовною схемою. Один із варіантів структурної схеми такого пристрою наведено на рис. 8.4.1.

Тривалість формованого таким пристроєм часового інтервалу визначається параметрами зовнішнього RC -кола. За активним значенням сигналу $U_{вх}$ RS -тригер устанавлюється в одиничний стан, що приводить до розмикання перемикача $S1$. Починається заряд конденсатора C зовнішнього кола, який задає час. У момент, коли

напруга на конденсаторі досягає рівня опорної напруги $U_{оп}$, відбувається спрацювання компаратора $DA1$ і його вихідний сигнал скидає RS -тригер. Перемикач $S1$ при цьому замикається і конденсатор C розряджається.



Рис. 8.4.1

Даний однотактний таймер, побудований за описаною схемою, може формувати на виході тільки одиничні імпульси. Для забезпечення можливості формування послідовності імпульсів схему пристрою потрібно доповнити іншим компаратором.

Означення 8.4.3. Багатотактним таймером називають таймер з убудованим лічильником, розробленим для формування імпульсів наднизької частоти з тривалістю імпульсу до кількох десятків годин.

Дані таймери можна поділити на дві підгрупи:

1. **Програмувальні таймери**, в яких часовий інтервал задається програмним способом. У найпростішому випадку це здійснюється встановленням на виводах лічильника зовнішніх перемичок;
2. **Спеціалізовані таймери**, лічильник яких має жорстко заданий коефіцієнт перерахування.

Структурна схема багатоактного таймера наведена на рис. 8.4.2.

Із рис. 8.4.2 випливає, що багатоактний таймер у своїй структурі містить однотактний таймер і двійковий лічильник, спільну роботу яких формує логічний блок. У багатотактному таймері фактично відбувається множення постійної часу зовнішнього RC -кола на модуль лічильника CT .