Міністерство освіти і науки України
Національний університет “ Львівська політехніка ”

Методичні вказівки до виконання лабораторних робіт
з курсу
“ Основи схемотехніки ч.2 ”

Львів - 2014

Лабораторна робота №1

Основи алгебри логіки

Мета роботи: Ознайомитись з основними елементарними логічними операціями

Лабораторна робота №2

Вивчення методів мінімізації логічних функцій

Мета роботи: Освоїти мінімізацію логічних функцій методами: безпосередніх перетворень, Квайна та карт Карно.

Теоретичні відомості

Мінімізація – це процес зведення логічної функції до такого виду, який буде більш простим і дешевим для її фізичної реалізації, тобто з меншим числом логічних елементів за рахунок зменшення числа логічних операцій, кількості змінних та зв‘язків між елементами.

Відомо кілька методів мінімізації, серед яких найбільш поширеними на практиці є такі:

- *безпосередніх перетворень;*
- *Квайна;*
- *карт Карно;*

Метод безпосередніх перетворень – це аналітичний метод спрощення логічних функцій за допомогою аксіом та законів булевої алгебри. Цей метод є досить ефективним для малої кількості змінних (не більше чотирьох).

Для мінімізації логічних функцій методом безпосередніх перетворень використовують такі закони бульової алгебри:

Закон склеювання

$$x_1x_2 \vee x_1\bar{x}_2 = x_1; \qquad (x_1 \vee x_2)(x_1 \vee \bar{x}_2) = x_1;$$

Закон поглинання

$$x_1(x_1 \vee x_2) = x_1; \qquad x_1 \vee x_1x_2 = x_1;$$
$$x_1(\bar{x}_1 \vee x_2) = x_1x_2; \qquad x_1 \vee \bar{x}_1x_2 = x_1 \vee x_2;$$

Закон тавтології

$$x_1 = x_1x_1 \cdot \ldots \cdot x_1; \qquad x_1 = x_1 \vee x_1 \vee \ldots \vee x_1;$$

У деяких випадках ефективніше мінімізувати інверсію логічної функції для отримання простішого виразу в МДНФ (мінімальна диз'юнктивна нормальна форма). Це може бути тоді коли логічна функція має переважаюче число одиниць, ніж нулів. Тоді, маючи МДНФ інверсної функції, для її технічної реалізації достатньо на виході побудованої схеми під'єднати інвертор для відновлення прямої функції.

Приклад. Мінімізувати логічну функцію:

$$y = \overline{x}_1\overline{x}_2x_3 \vee \overline{x}_1x_2\overline{x}_3 \vee \overline{x}_1x_2x_3 \vee \overline{x}_1\overline{x}_2\overline{x}_3 \vee x_1x_2\overline{x}_3$$

Розв'язання.

$$y = \overline{x}_1\overline{x}_2x_3 \vee \overline{x}_1x_2\overline{x}_3 \vee \overline{x}_1x_2x_3 \vee \overline{x}_1\overline{x}_2\overline{x}_3 \vee x_1x_2\overline{x}_3 =$$
$$= \overline{x}_1(\overline{x}_2x_3 \vee x_2\overline{x}_3 \vee x_2x_3 \vee \overline{x}_2\overline{x}_3) \vee x_2\overline{x}_3\underline{(x_1 \vee \overline{x}_1)} =$$
$$= \overline{x}_1(\overline{x}_2\underline{(x_3 \vee \overline{x}_3)} \vee x_2\underline{(\overline{x}_3 \vee x_3)}) \vee x_2\overline{x}_3 =$$
$$= \overline{x}_1\underline{(\overline{x}_2 \vee x_2)} \vee x_2\overline{x}_3 = \overline{x}_1 \vee x_2\overline{x}_3;$$

Зауважимо, що в даному прикладі крім вилучення доповнень застосована тавтологія мінтерму $\overline{x}_1x_2\overline{x}_3$.

Мінімізація логічних функцій методом Квайна

Для мінімізації даним методом логічна функція має бути зображена в УДНФ (удосконалена диз'юнктивна нормальна форма) або УКНФ (удосконалена кон'юнктивна нормальна форма). Можливість одержати мінімальну форму заданої логічної функції визначається теоремою Квайна:

- якщо в удосконаленій нормальній формі виконати всі операції склеювання, а потім всі операції поглинання, то в результаті буде одержано мінімальну форму.

Виконується в наступній послідовності:

1) функція, представлена в канонічній формі УДНФ (або УКНФ), приводиться до скороченої форми;

2) від скороченої форми переходять до мінімальної.

Щоб одержати скорочену форму з удосконаленої, кожний терм порівнюють з усіма іншими термами для виявлення серед них такої пари, яка б відрізнялася

лише одним аргументом Ax і $A\bar{x}$ або ($A \vee x$ і $A \vee \bar{x}$ для КНФ). Далі склеюються всі виявлені пари термів:

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

Отже, після склеювання маємо групу елементарних кон‘юнкцій - імплікант (або елементарних диз‘юнкцій - імпліцент) (n-1) – го рангу, з якою також, якщо це можливо, проводиться процедура склеювання для виявлення кон‘юнкцій (диз‘юнкцій) (n-2) –го рангу і т. д.

Потім проводиться операція поглинання на основі рівності $A \vee Ax = A(1 \vee x) = A$ (або $A \cdot (A \vee x) = A$).

Таким чином, отримується скорочена форма заданої логічної функції, яка у загальному випадку не є мінімальною, бо деякі її прості імпліканти[1] можуть бути покриті іншими імплікантами і тому їх можна вилучити.

Другий етап мінімізації зі скороченої до мінімальної форми полягає у вилученні усіх зайвих імплікант (імпліцент), для цього використовують таблицю (матрицю) Квайна. Рядки цієї таблиці утворюють прості імпліканти, а стовбці – мінтерми, тобто конституети 1 початкової логічної функції. Якщо проста імпліканта є складовою частиною мінтерма (або мінтермів) то у відповідній клітинці таблиці ставиться позначка « * ».

Для утворення мінімальної форми прості імпліканти виписуються за такими принципами:

- безпосередньо виписуються ті прості імпліканти, на перетинах яких з будь-яким мінтермом є тільки одна позначка; це базисні імпліканти, які утворюють ядро бульової функції і які обов‘язково входять в МДНФ;
- із сукупності простих імплікант вибираються такі, які з мінімальним сумарним числом символів (змінних) покривають позначками стовбці імплікантної таблиці.

Приклад. Мінімізувати логічну функцію методом Квайна

[1] Імплікантою називається елементарний добуток, рівний «1» на одному або декількох наборах, де дана функція рівна «1», і рівний «0» на всіх наборах, де дана функція рівна «0». Імпліканта покриває один або декілька мінтермів даної бульової функції. Зазвичай, імпліканта - це результат склеювання відповідних мінтермів або імплікант.

$$y = \overline{x}_2 x_1 \overline{x}_0 \vee \overline{x}_2 \overline{x}_1 x_0 \vee x_2 \overline{x}_1 x_0 \vee x_2 x_1 \overline{x}_0 \vee x_2 x_1 x_0;$$

Розв‘язання. Оскільки функція зображена в УДНФ, приступаємо до пошуку простих імплікант. Шукаємо пари мінтерм, що склеюються:

(1-4) - $x_1 \overline{x}_0$;

(2-3) - $\overline{x}_1 x_0$;

(3-5) - $x_2 x_0$;

(4-5) - $x_2 x_1$;

Будуємо скорочену ДНФ з одержаних простих імплікант:

$$y = x_1 \overline{x}_0 \vee \overline{x}_1 x_0 \vee x_2 x_0 \vee x_2 x_1;$$

Якщо в скоророчeній формі неможна виконати операції склеювання і поглинання, тоді будуємо імплікантну таблицю і досліджуємо можливість вилучення деяких простих імплікант.

	$\overline{x}_2 x_1 \overline{x}_0$	$\overline{x}_2 \overline{x}_1 x_0$	$x_2 \overline{x}_1 x_0$	$x_2 x_1 \overline{x}_0$	$x_2 x_1 x_0$
$x_1 \overline{x}_0$	*			*	
$\overline{x}_1 x_0$		*	*		
$x_2 x_0$			*		*
$x_2 x_1$				*	*

Таким чином, для даної логічної функції можливі дві тупикові форми:

$$y = x_1 \overline{x}_0 \vee \overline{x}_1 x_0 \vee x_2 x_0;$$

$$y = x_1 \overline{x}_0 \vee \overline{x}_1 x_0 \vee x_2 x_1;$$

Обидві тупикові форми мають однакову кількість членів, і будь-яка з них може бути вибрана як мінімальна ДНФ (МДНФ).

Мінімізація логічної функції методом карти Карно

Розглянемо візуальний метод мінімізації булевих функцій за допомогою карт (діаграм) Карно, який є одним з найбільш зручних методів при невеликому числі змінних (до 6). Цей метод був розроблений в 1953 р. американським вченим Морісом Карно.

Карта Карно є координатним способом представлення булевих функцій. При цьому способі завдання таблиця істинності функції представляється у вигляді координатної карти станів, яка містить 2^n комірок (n - кількість змінних). Змінні функції розбиваються на дві групи так, що одна група визначає координати стовпця карти, а інша - координати рядка. При такому методі побудови кожна комірка визначається значеннями змінних, які відповідають певному двійковому набору. Усередині кожної комірки карти Карно ставиться значення функції на цьому наборі. Змінні в рядках і стовпцях розташовуються так, щоб сусідні комірки карти Карно розрізнялися тільки в одному розряді змінної, тобто були сусідніми. Тому значення змінних в стовпцях і в рядках карти записані кодом Грея. Такий спосіб представлення дуже зручний для наочності при мінімізації булевих функцій.

Правила мінімізації з використанням карт Карно:

1. У карті Карно групи одиниць (для ДНФ) і групи нулів (для КНФ) необхідно обвести контурами. Всередині контура повинні знаходиться тільки однойменні значення функції. Цей процес відповідає операції склеювання або знаходження імплікант цієї функції.

2. Кількість об‘єднанних комірок всередині контура має бути кратною степені двійки (1, 2, 4, 8, 16..).

3. При об‘єднанні в контур крайні рядки (стовбці) карти (верхні і нижні, ліві і праві), а також кутові комірки, вважаються сусідніми (для карт до 4-х змінних).

4. Кожен контур повинен включати (об‘єднувати) максимально можливу кількість комірок. В цьому випадку він відповідатиме простій імпліканті.

5. Усі одиниці (нулі) в карті (навіть поодинокі) мають бути охоплені контурами. Будь-яка одиниця (нуль) може входити в контури довільну кількість разів.

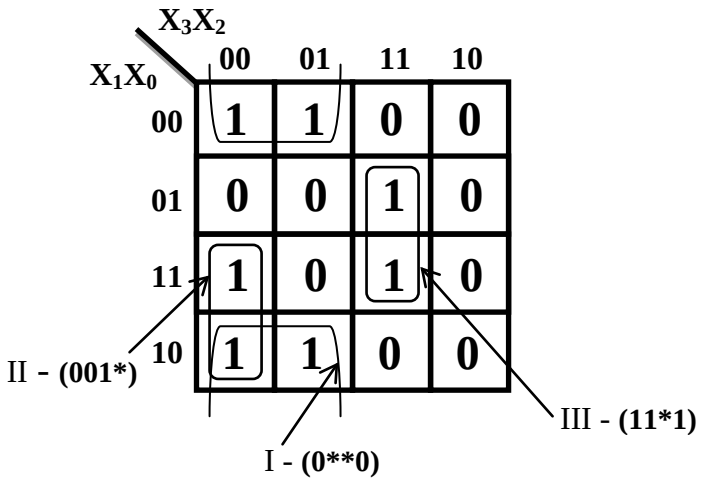
6. Безліч контурів, що покривають усі «1» («0») функції утворюють тупикову ДНФ (КНФ). Метою мінімізації є знаходження мінімальної з безлічі тупикових форм.

7. Елементарній кон'юнкції (диз'юнкції), яка відповідає одному контуру, залишаються тільки ті змінні, значення яких не змінюється всередині обведеного контура. Змінні бульової функції входять в елементарну конюнкцію (для значень функції в «1») без інверсії, якщо їх значення на відповідних координатах дорівнює «1» і з інверсією - якщо «0». Для значень булевої функції, рівних «0», записуються елементарні дизюнкції, куди змінні входять без інверсії, якщо їх значення на відповідних координатах дорівнює «0» і з інверсією - якщо «1».

Приклад 1 мінімізувати логічну функцію методом карти Карно:

$$y = \overline{x}_3\overline{x}_2\overline{x}_1\overline{x}_0 \vee \overline{x}_3\overline{x}_2x_1\overline{x}_0 \vee \overline{x}_3\overline{x}_2x_1x_0 \vee \overline{x}_3x_2\overline{x}_1\overline{x}_0 \vee \overline{x}_3x_2x_1\overline{x}_0 \vee x_3x_2\overline{x}_1x_0 \vee x_3x_2x_1x_0;$$

Щоб отримати МДНФ охоплюємо комірки що містять логічну 1, областями прямокутної форими. Області повинні містити 2^k комірок, де k - ціле число (2^k = 1; 2; 4; 8;…). Для кожної області складаємо комбінацію, в якій розряди, що відрізняються, позначаємо символом (*).



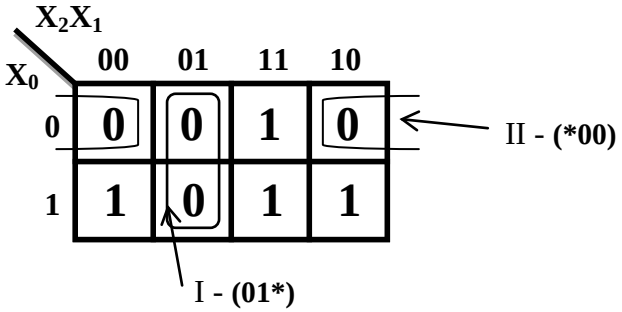
При мінімізації логічної функції отримуємо три області. Першій області відповідає набір (0**0), другій області - набір (001*) а третій області набір (11*1) . Отже, мінімальна ДНФ (МДНФ) запишеться у вигляді

$$y = \overline{x}_3\overline{x}_0 \vee \overline{x}_3\overline{x}_2x_1 \vee x_3x_2x_0;$$

Приклад 2 мінімізувати логічну функцію задану в УКНФ методом карти Карно:

$$y = (x_2 \vee x_1 \vee x_0)(x_2 \vee \overline{x}_1 \vee x_0)(x_2 \vee \overline{x}_1 \vee \overline{x}_0)(\overline{x}_2 \vee x_1 \vee x_0);$$

Щоб отримати мінімальну КНФ в карті Карно аналогічними прямокутними областями охоплюються нульові клітини, і також записуються набори, які відповідають охопленим областям



Для отримання диз'юнкцій, складових МКНФ, змінні позначають через інверсії наборів областей. Першій області відповідає набір (01*), диз'юнкція $(x_2 \vee \overline{x}_1)$ другій області - набір (*00), диз'юнкція $(x_1 \vee x_0)$. МКНФ логічної функції запишемо у вигляді

$$y = (x_2 \vee \overline{x}_1)(x_1 \vee x_0);$$

Завдання:

1. Мінімізувати задану логічну функцію трьома методами.
2. Побудувати удосконалену та мінімальну ДНФ (КНФ) форми графічно в середовищі Proteus.
3. Провести моделювання в середовищі Proteus функцій згідно варіанту.

Перелік варіантів завдань:

$$y_0 = \overline{x}_3\overline{x}_2x_1\overline{x}_0 \vee \overline{x}_3x_2x_1\overline{x}_0 \vee x_3\overline{x}_2\overline{x}_1\overline{x}_0 \vee x_3\overline{x}_2x_1\overline{x}_0 \vee x_3x_2\overline{x}_1\overline{x}_0 \vee x_3x_2x_1\overline{x}_0;$$

$$y_1 = \overline{x}_3\overline{x}_2\overline{x}_1\overline{x}_0 \vee \overline{x}_3\overline{x}_2x_1\overline{x}_0 \vee x_3\overline{x}_2\overline{x}_1\overline{x}_0 \vee x_3\overline{x}_2x_1\overline{x}_0 \vee x_3x_2\overline{x}_1x_0 \vee x_3x_2x_1x_0;$$

$$y_2 = \overline{x}_3x_2\overline{x}_1\overline{x}_0 \vee \overline{x}_3x_2\overline{x}_1x_0 \vee x_3\overline{x}_2\overline{x}_1x_0 \vee x_3\overline{x}_2x_1x_0 \vee x_3x_2x_1x_0;$$

$$y_3 = \overline{x}_3\overline{x}_2\overline{x}_1x_0 \vee \overline{x}_3\overline{x}_2x_1\overline{x}_0 \vee x_3\overline{x}_2\overline{x}_1x_0 \vee x_3\overline{x}_2x_1\overline{x}_0 \vee x_3x_2\overline{x}_1\overline{x}_0;$$

$$y_4 = \overline{x}_3\overline{x}_2\overline{x}_1x_0 \vee \overline{x}_3x_2\overline{x}_1x_0 \vee \overline{x}_3x_2x_1x_0 \vee x_3\overline{x}_2\overline{x}_1\overline{x}_0 \vee x_3\overline{x}_2x_1\overline{x}_0 \vee x_3x_2x_1x_0;$$

$$y_5 = \overline{x}_3\overline{x}_2\overline{x}_1\overline{x}_0 \vee \overline{x}_3\overline{x}_2\overline{x}_1 x_0 \vee \overline{x}_3 x_2 x_1\overline{x}_0 \vee \overline{x}_3 x_2 x_1 x_0 \vee x_3\overline{x}_2 x_1\overline{x}_0 \vee x_3 x_2 x_1 x_0;$$

$$y_6 = \overline{x}_3\overline{x}_2\overline{x}_1 x_0 \vee \overline{x}_3\overline{x}_2 x_1 x_0 \vee \overline{x}_3 x_2\overline{x}_1 x_0 \vee \overline{x}_3 x_2 x_1 x_0 \vee x_3\overline{x}_2 x_1 x_0 \vee x_3 x_2\overline{x}_1 x_0;$$

$$y_7 = \overline{x}_3\overline{x}_2 x_1\overline{x}_0 \vee \overline{x}_3 x_2\overline{x}_1\overline{x}_0 \vee \overline{x}_3 x_2 x_1\overline{x}_0 \vee x_3 x_2\overline{x}_1\overline{x}_0 \vee x_3 x_2\overline{x}_1 x_0;$$

$$y_8 = \overline{x}_3 x_2\overline{x}_1\overline{x}_0 \vee \overline{x}_3 x_2\overline{x}_1 x_0 \vee \overline{x}_3 x_2 x_1\overline{x}_0 \vee \overline{x}_3 x_2 x_1 x_0 \vee \overline{x}_3\overline{x}_2 x_1 x_0 \vee x_3\overline{x}_2 x_1 x_0 \vee x_3 x_2 x_1 x_0;$$

$$y_9 = \overline{x}_3\overline{x}_2\overline{x}_1\overline{x}_0 \vee x_3\overline{x}_2\overline{x}_1\overline{x}_0 \vee x_3\overline{x}_2 x_1\overline{x}_0 \vee x_3 x_2\overline{x}_1\overline{x}_0;$$

$$y_{10} = (x_3 \vee x_2 \vee x_1 \vee x_0)\cdot(x_3 \vee x_2 \vee \overline{x}_1 \vee x_0)\cdot(x_3 \vee \overline{x}_2 \vee x_1 \vee x_0)\cdot(x_3 \vee \overline{x}_2 \vee \overline{x}_1 \vee x_0)\times$$
$$\times(\overline{x}_3 \vee x_2 \vee x_1 \vee x_0)\cdot(\overline{x}_3 \vee x_2 \vee x_1 \vee \overline{x}_0);$$

$$y_{11} = (x_3 \vee \overline{x}_2 \vee x_1 \vee x_0)\cdot(x_3 \vee x_2 \vee \overline{x}_1 \vee \overline{x}_0)\cdot(\overline{x}_3 \vee x_2 \vee x_1 \vee x_0)\cdot(x_3 \vee x_2 \vee x_1 \vee x_0)\cdot(x_3 \vee \overline{x}_2 \vee \overline{x}_1 \vee x_0);$$

Лабораторна робота №3

Синтез комбінаційних пристроїв в базисі

Шеффера (І-НЕ) та Пірса (АБО-НЕ)

Мета роботи: Навчитися синтезувати комбінаційні пристрої в заданому базисі.

Теоретичні відомості

Завдання синтезу є неоднозначним. Воно залежить від логічних елементів, з яких буде зібрано пристрій. Навіть при заданій елементній базі буває можливість представити логічний вираз в різній формі і, відповідно, по-різному побудувати схему комбінаційного пристрою. Причому, будь-яка з схем однаково виконує задані функції.

Синтез комбінаційного пристрою здійснюється в наступній послідовності:

1. Функції, представлені в довільній формі (найчастіше в табличній), записують у вигляді логічного виразу ДДНФ або ДКНФ.
2. Проводиться мінімізація логічних функцій будь-яким методом.
3. Логічні функції переводяться в заданий базис, відповідний обмеженням на елементну базу.
4. Будується функціональна схема комбінаційного пристрою.

Перші два етапи синтезу детально розглянуто в попередніх роботах. Функціонально повна система (базис) – це сукупність логічних елементів, яка дозволяє реалізувати будь-яку логічну схему довільної складності.

Для побудови складної логічної функції немає необхідності використовувати всі елементарні функції. Можна обмежити набір елементарних функцій, виключивши з нього ті елементи, які можна виразити через інші. Послідовно виключаючи з базису функції, получають мінімальний базис. Під мінімальним базисом розуміють такий набір функцій, виключення з якого будь-якої функції перетворює повну систему на неповну.

Можливі різні базиси, що відрізняються один від одного числом функцій, що входять в них, і видом цих функций. Вибір того або іншого базису для побудови логічних пристроїв обумовлений тим, наскільки економічно вигідно і технічно просто виконати

елементи, які реалізують функції, що входять в базис, і у весь логічний пристрій в цілому.

Через три логічні функції інверсії (НЕ), кон‘юнкції (І), диз‘юнкції (АБО) можна виразити будь-яку елементарну функцію і побудувати будь-якої складності логічний пристрій. Набір з трьох функцій (НЕ, І, АБО) є базисом. Проте, базис (НЕ, І, АБО) не являється мінімальним. З нього можна виключити одну з функций І чи АБО. Набори (НЕ, І), а також (НЕ, АБО) з двох функцій будуть мінімальними базисами.

Досить зручно технічно реалізувати на мікросхемах логічні елементи, що поєднують в собі вказані функції. Це елементи І-НЕ (штрих Шеффера) і АБО-НЕ (стрілка Пірса). Кожен з елементів (І-НЕ), (АБО-НЕ) окремо є функціонально повним базисом.

Перевагою базисів з однієї логічної функції (І-НЕ) чи (АБО-НЕ) є те, що весь логічний пристрій побудований тільки на однотипних логічних елементах. Отримана логічна схема, володіє регулярною структурою. Необхідно тільки здійснити комутацію однакових логічних елементів. Базиси на логічних елементах (І-НЕ), (АБО-НЕ) широко используются при проектуванні пристроїв, зручні для реализації у великих інтегральних схемах. Зменшення номенклатуры до одного типу, таким чином, полегшує проектування пристроїв.

Для переходу з базису (НЕ, І, АБО) до (І-НЕ), (АБО-НЕ) використовують такі закони булевої алгебри:

-теореми де-Моргана

$$\overline{x_1 \vee x_2 \vee \ldots \vee x_n} = \overline{x}_1\overline{x}_2\ldots\overline{x}_n; \qquad \overline{x_1 x_2 \ldots x_n} = \overline{x}_1 \vee \overline{x}_2 \vee \ldots \vee \overline{x}_n;$$

-закони тафтології

$$x_1 = x_1 x_1 \cdot \ldots \cdot x_1; \qquad x_1 = x_1 \vee x_1 \vee \ldots \vee x_1;$$

$$\overline{x_1} = \overline{x_1 x_1 \cdot \ldots \cdot x_1}; \qquad \overline{x_1} = \overline{x_1 \vee x_1 \vee \ldots \vee x_1};$$

-закон подвійної інверсії

$$\overline{\overline{x}} = x;$$

Синтез комбінаційного пристрою в базисі Шеффера (І-НЕ)

Для синтезу комбінаційного пристрою в базисі Шеффера (І-НЕ) отримують мінімальну диз'юнктивну нормальну форму (МДНФ). Подальші перетворення проводять на основі вище згаданих законів.

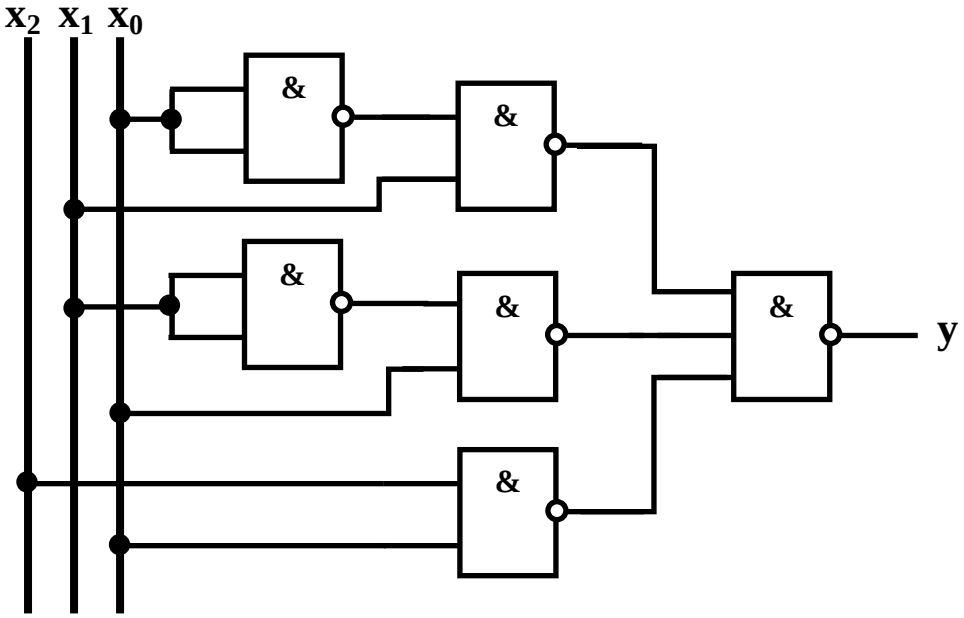
Приклад 1: зобразити функцію $y = x_1\bar{x}_0 \vee \bar{x}_1 x_0 \vee x_2 x_0$ в базисі Шеффера.

Двічі інвертуємо вираз $y = \overline{\overline{x_1\bar{x}_0 \vee \bar{x}_1 x_0 \vee x_2 x_0}};$

Застосовуємо формулу де Моргана $y = \overline{\overline{x_1\bar{x}_0} \cdot \overline{\bar{x}_1 x_0} \cdot \overline{x_2 x_0}};$

Застосовуємо закон тафтології $y = \overline{\overline{x_1 x_0 x_0} \cdot \overline{x_1 x_1 x_0} \cdot \overline{x_2 x_0}};$

Будуємо функціональну схему



Синтез комбінаційного пристрою базисі Пірса (АБО-НЕ)

При синтезі в базисі Пірса (АБО-НЕ) використовують такий же підхід, але є деякі особливості. У цьому випадку має бути отримана мінімальна кон'юнктивна нормальна форма (МКНФ).

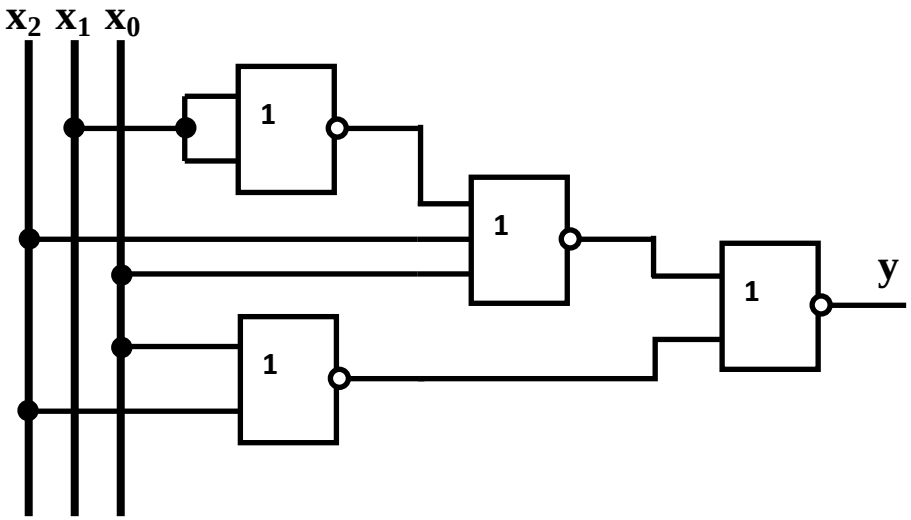
Приклад 1: зобразити функцію $y = (x_2 \vee \bar{x}_1 \vee x_0)(x_2 \vee x_0)$ в базисі Пірса.

Двічі інвертуємо вираз $y = \overline{\overline{(x_2 \vee \bar{x}_1 \vee x_0)(x_2 \vee x_0)}};$

Застосовуємо формулу де Моргана $y = \overline{\overline{x_2 \vee \bar{x}_1 \vee x_0} \vee \overline{x_2 \vee x_0}};$

Застосовуємо закон тафтології $y = \overline{\overline{x_2 \vee \overline{x_1 \vee x_1} \vee x_0} \vee \overline{x_2 \vee x_0}};$

Будуємо функціональну схему



Завдання:

4. Синтезувати задану логічну функцію в базисі Пірса та Шеффера.
5. Провести моделювання в середовищі Proteus.

Перелік варіантів завдань:

$y_0 = \sum(0,3,4,5,7,8,10,11,14,15);$

$y_1 = \sum(0,2,4,6,8,9);$

$y_2 = \prod(1,5,7,8,10,15);$

$y_3 = \prod(4,5,6,7,3,11,15);$

$y_4 = \sum(3,7,8,9,11,15);$

$y_5 = \prod(1,2,9,10,12,13);$

$y_6 = \sum(0,2,8,13,15,10);$

$y_7 = \sum(0,2,6,7,8,10,11);$

$y_8 = \prod(2,4,6,12,13);$

$y_9 = \sum(0,3,4,5,7,8,10,11,14,15);$

$y_{10} = \prod(1,3,5,7,11,13);$

$y_{11} = \sum(1,2,3,4,5,9,11,12,13,14,15);$

$y_{12} = \prod(0,8,9,12);$

Лабораторна робота №4

Дослідження комбінаційних пристроїв

Мета:Вивчення призначення та принципів роботи КП. Навчитися реалізовувати та досліджувати функціональні модулі на основі КП.

Теоретичні відомості

Комбінаційні пристрої

Комбінаційним пристроєм (КП) називається цифровий пристрій, в якого значення вихідних сигналів залежать лише від значень сигналів на його входах у даний момент часу і не залежить від значень сигналів, які діяли у попередній момент часу.

До них належать:

- Перетворювачі кодів
- Шифратори (дешифратори)
- Мультилексори (демультилексори)
- Суматори
- Цифрові компаратори

Синтез цифрового автомата (в тому числі і комбінаційного) полягає у побудові такого автомата, який реалізує наперед задану довільним чином функції «вхід-вихід».

Етапи синтезу

1. Формалізований запис завдання алгоритму функціонування КП;
2. Оцінка розмірності задачі (тобто числа змінних);
3. Мінімізація логічних функцій;
4. Перетворення мінімізованих логічних функцій у раціональну для їх реалізації форму до заданого або вибраного базису;
5. Побудова структурної схеми КП;
6. Вибір для реалізації принципової схеми конкретної серії мікросхем;
7. Розробка принципової схеми КП;
8. Перевірка працездатності спроектованого КП на логічні тести.

Шифратори та дешифратори

Шифратор (Coder: CD)

Призначений для перетворення ,кодування або шифрування , алфавітно-цифрової інформації, що подана унітарним ***n*** -розрядним кодом, у еквівалентний двійковий m-розрядний код.

Особливістю унітарного коду є активний /збуджений/ стан тільки однієї змінної вхідного набору$\{X_{n-1}\ldots X_1\mathbf{X_0}\}$,порядковий номер ***i*** якої саме й підлягвє шифруванню (кодуванню). Отже, шифратор n -

m - це перетворювач унітарного коду "1 з ***n*** " у двійковий, як правило, паралельний код, у якого число виходів ***m*** однозначно зв'язане з числом входів n як 2^m .Якщо $n=2^m$ що означає використання повного набору вихідних двійкових комбінаційY_i ,такий шифратор називають повним. Наприклад, шифратор 8-3 є повним, бо він реалізує повний набір можливих комбінацій змінних X_i (n = 8) у повний вихідний набір Y_i ($m - 2$) як $2^3 = 8$.

Наприклад,шифратор 8-3 (вісім входів,три виходи)

У *неповному шифраторі* число входів ***n*** не відповідає числу всіх можливих вихідних комбінацій 2^m , причому завжди $n < 2^m$, що відповідно утворює певне число невикористаних вихідних наборів. Прикладом неповного шифратора, який найчастіше зустрічається на практиці, є шифратор 10-4, що використовується для кодування десяткових чисел у двійково-десятковий код ДДК (8-4-2-1). Такий шифратор можна застосовувати для кодування десяткових символів (0...9), наприклад, з клавіатури пульта керування.

Синтез шифраторів (8-3)

Таблиця істинності:

Десяткове число	X0	X1	X2	X3	X4	X5	X6	X7	Y2	Y1	Y0
0	1	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	1
2	0	0	1	0	0	0	0	0	0	1	0
3	0	0	0	1	0	0	0	0	0	1	1
4	0	0	0	0	1	0	0	0	1	0	0
5	0	0	0	0	0	1	0	0	1	0	1
6	0	0	0	0	0	0	1	0	1	1	0
7	0	0	0	0	0	0	0	1	1	1	1

Ця таблиця істинності відповідає системі логічних функцій:

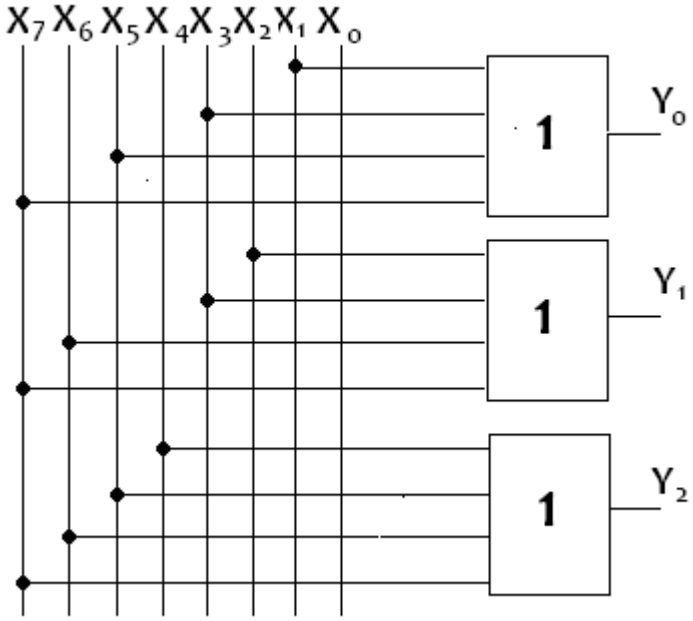
$$Y_0 = X_1 \vee X_3 \vee X_5 \vee X_7 = \overline{\overline{X_1} \cdot \overline{X_3} \cdot \overline{X_5} \cdot \overline{X_7}}$$
$$Y_1 = X_2 \vee X_3 \vee X_6 \vee X_7 = \overline{\overline{X_2} \cdot \overline{X_3} \cdot \overline{X_6} \cdot \overline{X_7}}$$
$$Y_2 = X_4 \vee X_5 \vee X_6 \vee X_7 = \overline{\overline{X_4} \cdot \overline{X_5} \cdot \overline{X_6} \cdot \overline{X_7}}$$

З отриманої системи функцій видно, що шифратор 8-3 легко реалізувати у базисі НЕ-4-АБ0 /рис. 4.6/, а для інверсних входів - на НЕ- 4І-НЕ, де $У_0$- вихід молодпого розряду з вагою 2^0 , $У_1$- вихід з вагою 2^1 і $У_2$- вихід старшого розряду з вагою 2^2.

Схему шифраторf можна будувати за так званим лінійним принципом - коли всі НЕ-4-АБ0 під'єднують до однієї спільної шини /лінії/ так, як показано на рис. нижче.

Для лінійних шифраторів характерна незадіяна змінна X_o /див. табл. 4.1/, Це означає, що при

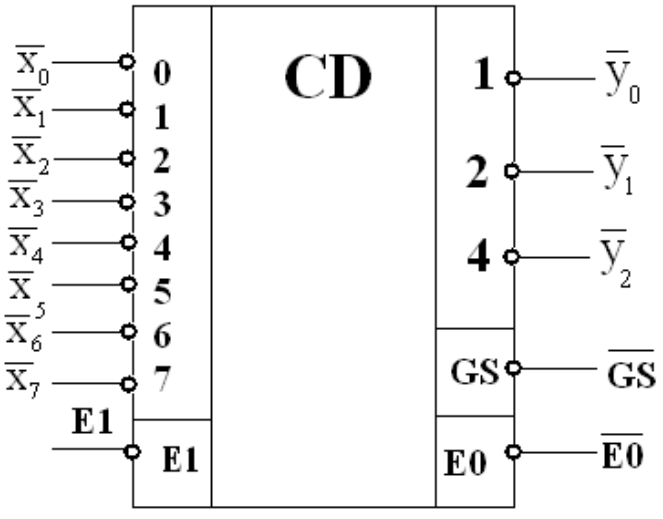
будь-якому сигналі на вході X_o на виході шифратора не буде жодних змін. Однак така ситуація на практиці завжди враховується і тому передбачається обов`язкова присутність активного входу.



Неповний шифратор реалізується аналогічно. Але у нього є невикористані вихідні набори. Наприклад,у шифраторі 10-4 число невикористаних входів 16-10=6 і невикористовувані вихідні комбінації Y_i = 1010,1011,1100,1101,1110,1111.

Меншу швидкодію мають шифратори,що побудовані за принципом використання однотипних, наприклад, двовходових ЛЕ 2І–НЕ,структурна схема якого нагадує піраміду.

<u>У пріоритетного шифратора</u> вихідний код завжди відповідає тому активному входу ,який має найбільший номер набору. Так у випадку одночасно активних входів X_0 ,X_3 ,X_6 на виході буде код 110 (6)



<u>Приклад К555ИВ1</u>- пріоритетний шифратор 8-3 перетворює сигнал низького рівня на одному з восьми інформаційних входів ($\overline{X}_0 \ldots \overline{X}_7$) у трирозрядний обернений двійковий код ($\overline{Y}_0, \overline{Y}_1, \overline{Y}_2$)

E1-вхід дозволу роботи (0);

E0-вихід сигналу дозволу, який при E0=0 фіксує одиниці на всіх входах;

GS (Grup Signal) - вихід групового сигналу;

GS=0 при наявності на входах хоча б одного нуля.

(GS і E0 дозволяють реалізувати каскадування шифратора)

К555ИВ3- шифратор 10-4

К555ИВ2-шифратор 8-3 – з тристановими виходами.

Шифратори n-mвикористовуються для перетворення буквено-цифрових символів, десяткових, вісімкових і шістнадцяткових чисел у двійкову форму.

Пріоритетні шифратори використовуються для побудови ЦАП і АЦП та у мікропроцесорній техніці.

Дешифратор (Decoder:DC)

Призначений для розпізнавання (дешифрації) числа, яке подане позиційним n-розрядним двійковим кодом .

Найчастіше дешифратор n-m виконує функцію перетворення двійкового коду в унітарний код «1 з m», тобто функцію, обернену дії шифратора.

$m=2^n$, де m- порядковий номер виходу Y_i дешифратора у попередній таблиці істинності (Y замінити на X і навпаки)

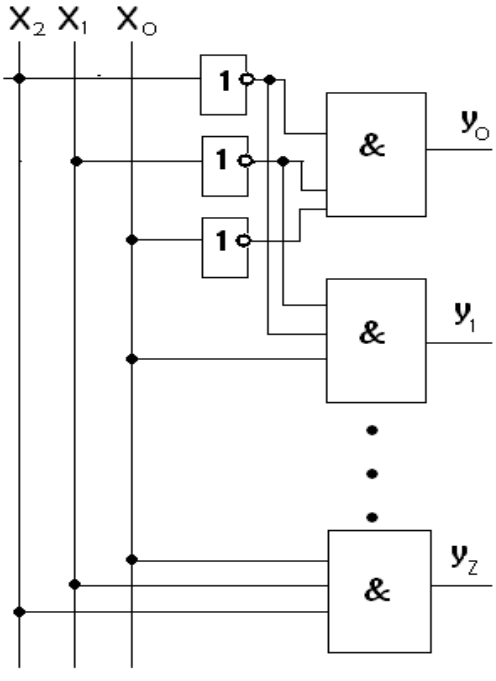
Описується системою булевих функцій

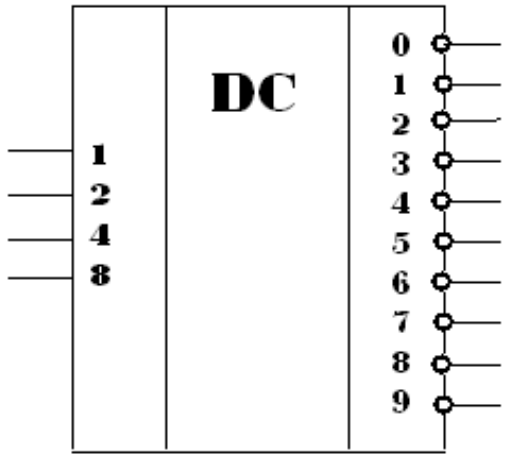
$$Y_0 = \overline{X_{n-1}}\ \overline{X_{n-2}} \dots \overline{X_1}\ \overline{X_0} = \overline{X_{n-1} \vee X_{n-2} \vee \dots X_1 \vee X_0}$$
:
$$Y_1 = \overline{X_{n-1}}\ \overline{X_{n-2}} \dots \overline{X_1}\ X_0 = \overline{\overline{X_{n-1} \vee X_{n-2} \vee \dots X_1 \vee \overline{X_0}}}$$
$$Y_2 = \overline{X_{n-1}}\ \overline{X_{n-2}} \dots X_1\ \overline{X_0} = \overline{\overline{X_{n-1} \vee X_{n-2} \vee \dots \overline{X_1} \vee X_0}}$$
.
$$Y_{m-2} = X_{n-1}\ X_{n-2} \dots X_1\ \overline{X_0} = \overline{\overline{\overline{X_{n-1}} \vee \overline{X_{n-2}} \vee \dots \overline{X_1} \vee X_0}}$$
$$Y_{m-1} = X_{n-1}\ X_{n-2} \dots X_1\ X_0 = \overline{\overline{\overline{X_{n-1}} \vee \overline{X_{n-2}} \vee \dots \overline{X_1} \vee \overline{X_0}}}$$

Наприклад, ДШ 3-8 описується

$$Y_0 = \overline{X_2}\,\overline{X_1}\,\overline{X_0} = (000) \qquad Y_4 = X_2\,\overline{X_1}\ \overline{X_0} = (100)$$
$$Y_1 = \overline{X_2}X_1X_0 = (001) \qquad Y_5 = X_2\ \overline{X_1}\,X_0 = (101)$$
$$Y_2 = \overline{X_2}\,X_1\,\overline{X_0} = (010) \qquad Y_6 = X_2X_1\,\overline{X_0} = (110)$$
$$Y_3 = \overline{X_2}\ X_1\,X_0 = (011) \qquad Y_7 = X_2X_1X_0 = (111)$$

Структурна схема ДШ 3→8:





По аналогії з шифраторами дешифратори (ДШ) бувають повні і неповні.

К155ИД3, К155ИД7 - повні ДШ

К155ИД1, К176ИД1, К555ИД6, К555ИД10, К561ИД1 - неповні.

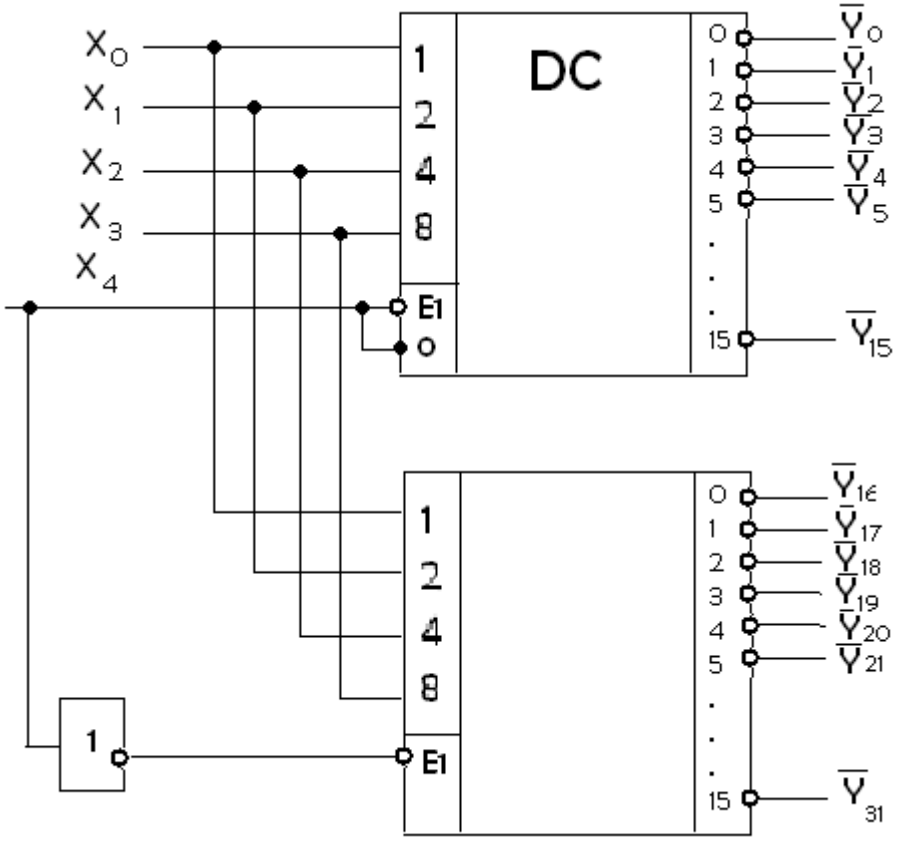
К155ИД1 (4-10)

З відкритим колектором (високовольтний U≤60В)

Застосовуються, наприклад, для керування роботою газорозрядних індикаторів.

Для збільшення кількості виходів використовується каскадування .

Коли потрібно побудувати дешифратор на велику кількість виходів на базі дешифраторів з меншим числом виходів, застосовують принцип каскадування. Він полягає у тому, що дані входи дешифраторів розбивають довільним чином на групи, кожна з яких реалізує свою групу ло¬гічних функцій. При цьому всі дешифратори повинні бути керованими, тобто мати дозволяючі входи Е1 або вільний вхід старшого розряду.



Застосування ДШ:

- Широке застосування дешифратори мають у пристроях візуальної індикації десяткових цифр на світлових табло, що використовують світлодіоди, індикатори на рідких кристалах,

електролюмінесцентні чи електровакуумні прилади. Такі дешифратори випускають у вигляді мікросхем середнього ступеня інтеграції конкретного призначення.

Перетворювачі кодів

Вони призначені для перетворення одного різновиду цифрового коду в інший. Необхідність у таких перетворювачах для цифрових пристроїв пояснюється тим, що в деяких випадках технічно вигідніше і навіть точніше виконувати певні операції з допомогою інших кодів, не лише одним двійковим кодом. Різновидів цифрових кодів є дуже багато. Одним із них є код Грея.

Код Грея - система числення, в якій два сусідніх значення розрізняються тільки в одному розряді. Найбільш часто на практиці застосовується рефлексивний двійковий код Грея , хоча в загальному випадку існує нескінченна безліч кодів Грея для систем числення з будь-якою основою . У більшості випадків , під терміном « код Грея » розуміють саме рефлексивний бінарний код Грея .

Спочатку призначався для захисту від помилкового спрацьовування електромеханічних перемикачів. Сьогодні коди Грея широко використовуються для спрощення виявлення та виправлення помилок в системах зв'язку , а також у формуванні сигналів зворотного зв'язку в системах управління .

Перевід двійкового коду в код Грея і доповняльний код:

Десяткова цифра	Двійковий код	Доповняльний код	Код Грея
0	0000	0000	0000
1	0001	1111	0001
2	0010	1110	0011
3	0011	1101	0010
4	0100	1100	0110
5	0101	1011	0111
6	0110	1010	0101
7	0111	1001	0100
8	1000	1000	1100
9	1001	0111	1101
10	1010	0110	1111
11	1011	0101	1110
12	1100	0100	1010
13	1101	0011	1011
14	1110	0010	1001
15	1111	0001	1000

Перетворювачі кодів необхідні насамперед для технічної реалізації різних арифметичних операцій над двійковими числами, а також для вводу та виводу числової інформації з однієї системи числення в іншу.

На відміну від дешифратора перетворювач кодів може формувати довільні двійкові числа. Кожному вхідному набору ставиться у відповідність вихідний набір, а не унітарний код, як це має місце у дешифратора.

Мультиплексори і демультиплексори

Це КП, що призначені для комутації цифрових каналів під дією двійкового коду керуючих сигналів.

Мультиплексор (MUX)

Мультиплексор (ще інші назви - селектор, багатопозиційний комутатор, селектор-мультиплексор) – призначений для передачі (комутації) сигналів від одного з кількох інформаційних входів X_i на один вихід. Крім інформаційних має адресні входи a_j, двійковий код яких визначає номер інформаційного входу, який передається на вихід.

MUX має $m=2^n$ інформаційних і n адресних входів та один вихід.

Це КП, що призначені для комутації /цифрових каналів під дією двійкового коду керуючих сигналів. Згідно зі своїм призначенням мультиплексор реалізує логічну функцію

$$Y = E\bigvee_{i=0}^{n} x_i m(a)_i$$

де x_i. – вхідні інформаційні сигнали; $m(a)_i$ - мінтерми n адресних змінних

Для побудови мультиплексора $2^n \rightarrow 1$ потрібно мати багатовходовий ЛЕ типу І-АБО, який би забезпечував передачу (комутацію) з інформаційної шини даних одного з 2 сигналів, а для керування комутацією - дешифратор. На рис. нижче показана схема мультиплексора 4→1, що з допомогою n = 2 адресних сигналів a_1 і a_0 забезпечує вибір одного з $2^2 = 4$ даних x_i . Отже, згідно з /4.7/ логічна функція мультиплексора 4-І має вигляд

$$Y = x_0\bar{a}_1\bar{a}_0 \vee x_1\bar{a}_1 a_0 \vee x_2 a_1\bar{a}_0 \vee x_3 a_1 a_0$$

Р мультиплексорів з різним числом адресних входів, найчастіше з числом n = «2, 3, 4, ...

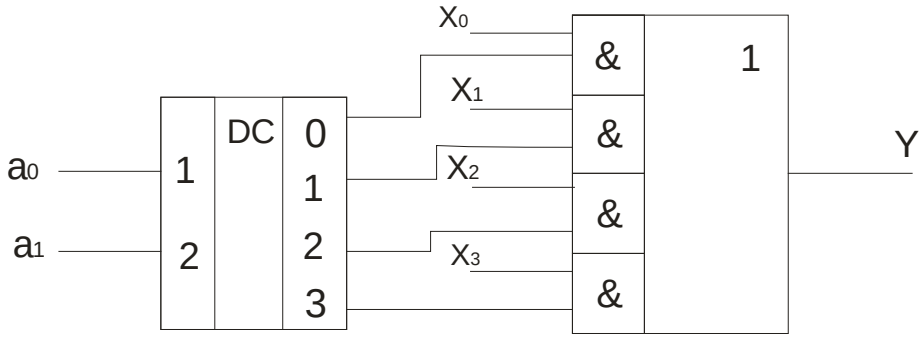


Рис. схема реалізації мультиплексора 4→1 на основі дешифратора

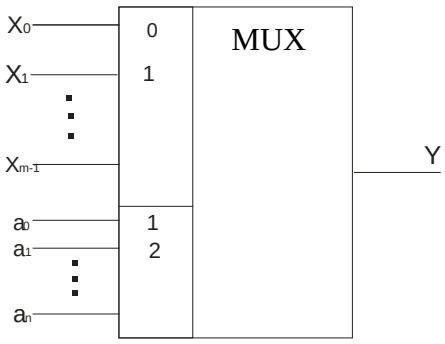


Рис. приклад MUX m → 1 ($2^n \rightarrow 1$)

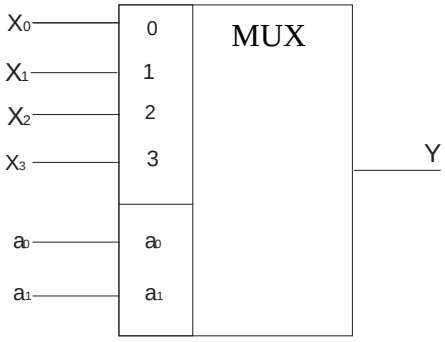


Рис. приклад MUX 4 → 1 ($2^2 \rightarrow 1$)

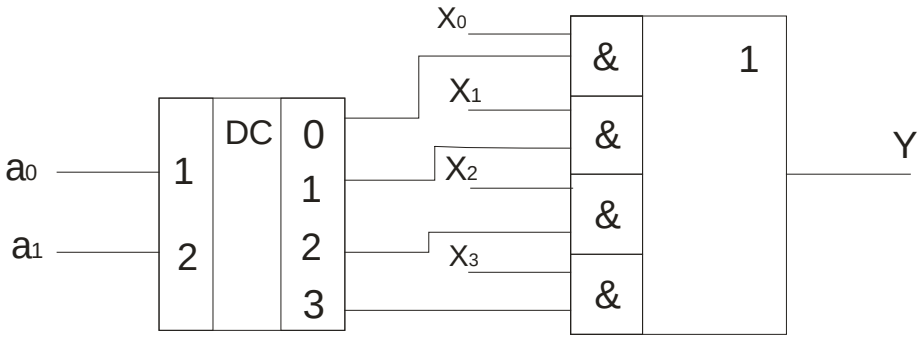
Таблиця істинності MUX 4 → 1 :

N входу	a_1	a_0	Y
0	0	0	X_0
1	0	1	X_1
2	1	0	X_2
3	1	1	X_3

Маючи таблицю істинності можемо знайти ДНФ.

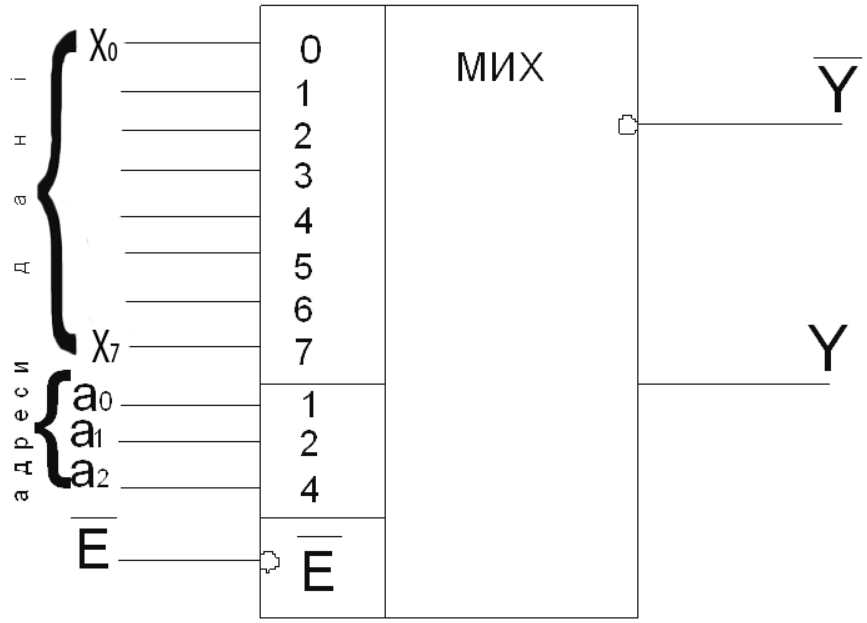
$$Y = x_0 \overline{a_0}\,\overline{a_1} \vee x_1 a_0 \overline{a_1} \vee x_2 \overline{a_0} a_1 \vee x_3 a_0 a_1$$

Тепер можемо скласти схему мультиплексора, в основі якого використаємо дешифратор:



Часто використовуваною є мікросхема мультиплексора MUX 8-1 К155КП7.

Вона має 8 входів, прямий і інверсний вихід, $\overline{E}$ - вхід дозволу (при Е=0) У=f(a) мультиплексор працює. Якщо на вхід дозволу подається логічна одиниця, на виході Y=0; $\overline{Y} = 1$;



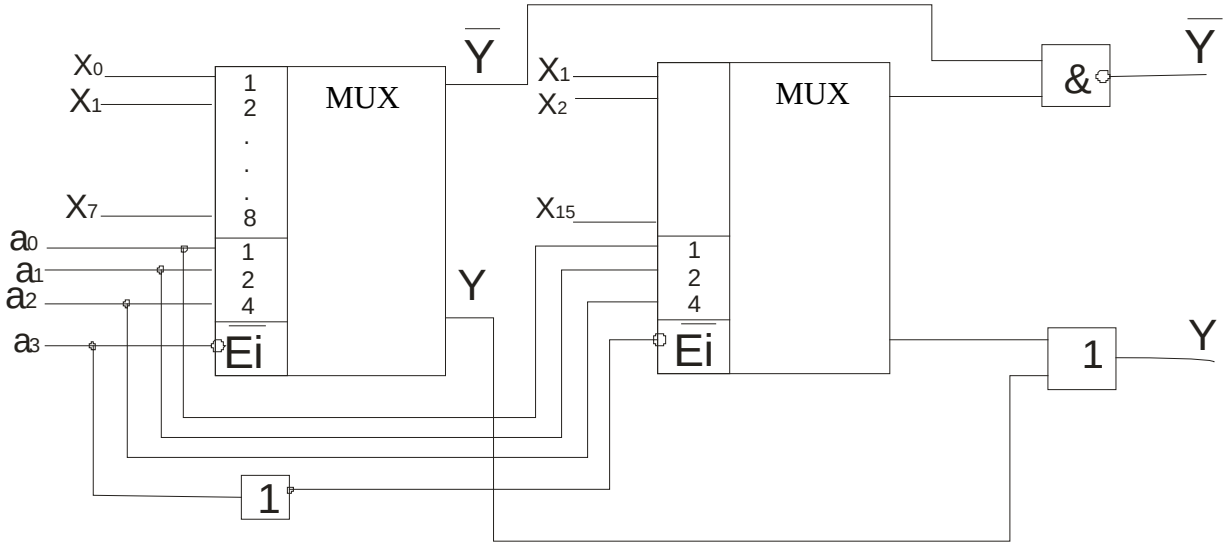
$$Y = \overline{E}(x_0 \overline{a_2}\,\overline{a_1}\,\overline{a_0} \vee x_1 \overline{a_2}\,\overline{a_1} a_0 \vee x_2 \overline{a_2} a_1 \overline{a_0} \vee \ldots \vee x_7 a_2 a_1 a_0)$$

(При: $\overline{E} = 1$: Y=0; $\overline{Y} = 1$; $\overline{E} = 0$ Y=f(a))

В реальному житті часто виникає потреба в збільшенні кількості входів мультиплексора.

Для комутування великої кількості входів застосовують принцип каскадування.

Приклад. Побудувати MUX 16-1 на базі мікросхеми 8-1.



За допомогою мультиплексорів можна реалізувати безліч різноманітних цифрових, а у деяких випадках цифро - аналогових схем. Наприклад, на базі мультиплексорів реалізуються: КП багатьох змінних, багатоканальні комутатори цифрових та аналогових сигналів, запам'ятовувальні пристрої, генератори послідовностей двійкових чисел, різні функціональні вузли тощо.

MUX може застосовуватися для перетворення паралельного коду в послідовний; а також для послідовного комутування сигналів на 2^n інформаційних входах.

Демультиплексор (DMX)

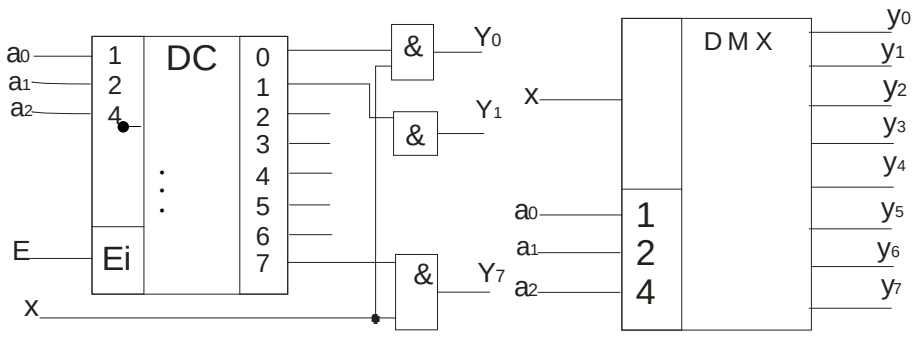
Призначений для виконання оберненої функції мультиплексора – передачу (комутацію) сигналу з єдиного інформаційного входу на один з 2^n виходів залежно від коду на n-керуючих входах.

Для початку побудуємо таблицю істинності

N	a_2	a_1	a_0	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	0	x	0	0	0	0	0	0	0
1	0	0	1	0	x	0	0	0	0	0	0
2	0	1	0	0	0	x	0	0	0	0	0
3	0	1	1	0	0	0	x	0	0	0	0
4	1	0	0	0	0	0	0	x	0	0	0
5	1	0	1	0	0	0	0	0	x	0	0
6	1	1	0	0	0	0	0	0	0	x	0
7	1	1	1	0	0	0	0	0	0	0	x

Схема

Можливий варіант побудови демультиплексора:



а) б)

Рис. а) побудова демультиплексора на основі дешифратора. б) схемне позначення демультиплексора 8 -> 1

Паралельний код, що керує роботою демультиплексора, подать на входи дешифратора, які стають адресними, а виходи останнього під"єднують до входів кон’юнкторів**.**

У загальному випадку демультиплексор реалізує логічну функцію виду $Y_i = XM_i(a)$.

Прикладами демультиплексорів є мікросхеми К155 ИД3, ИД6, ИД4, ИД7, ИД10. Демультиплексори можуть застосовуватися як перетворювач послідовного коду в паралельний.

Функціональну дуальність мультиплексора і демультиплексора зручно використати для передачі інформаційних двійкових сигналів (даних) на відстань, наприклад, по телефонних лініях зв"язку та кабелях. В такій системі звиязку функцію передавача-перетворювача паралельного коду в послідовний виконує мультиплексор, а функцію приймача-перетворювача послідовного коду у паралельний—демультиплексор. При наявності лінії керування обох перетворювачів забезпечується синхронна робота системи. Перевага такої системи передачі даних на відстань, незважаючи на її низьку швидкодію, полягає в економії затрат, які неминучі при паралельній передачі інформації,

Синтез комбінаційних пристроїв на дешифраторах та мультиплексорах

1. Синтез КП на базі дешифраторів.

На основі ЛЕ можна побудувати найрізноманітніші КП. Однак логічне проектування складних цифрових пристроїв та систем має певні функціональні обмеження і, крім цього, погіршує надійність, а то й швидкодію КП, бо із збільшенням числа ЛЕ проявляються всі властиві КП негативні явище. Виявляється, що деякі КП, крім своїх конкретних функцій, можуть успішно використовуватися для побудови інших більш чи менш складних КП. При цьому реалізація різноманітних цифрових пристроїв та систем на їх основі дозволяє поліпшити надійність, швидкодію і габаритні розміри, а у деяких випадках - зменшити потужність і вартість цих пристроїв та систем.

До функціональних елементів, на яких можна синтезувати різні комбінаційні пристрої, належать дешифратор і мультиплексор.

N	X_0	X_1	X_2	Y_1	Y_2
0	0	0	0	0	1
1	0	0	1	1	0
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	1
6	1	1	0	1	0
7	1	1	1	0	0

$$Y_1 = k_1 \vee k_3 \vee k_5 \vee k_6$$

$$Y_2 = k_0 \vee k_2 \vee k_4 \vee k_5$$

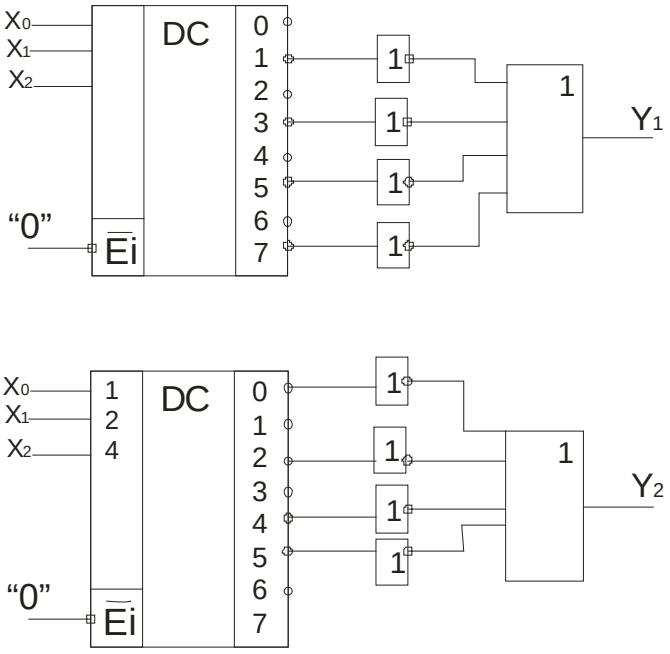


Рис. Схемна реалізація комбінаційного пристрою, заданого таблицею істинності

2. Синтез КП на мультиплексорі.

Він полягає у тому, що задавши інформаційному входу лог. 0 або лог. 1, можна виключити або включити у логічну функцію той чи інший мінтерм, який визначається адресним кодом. Таким чином, логічними рівнями можна вибрати потрібні для синтезу КП мінтерми для реалізації заданої логічної функції, а застосувавши каскадування, й систему функцій.

Отже, мультиплексор зручно застосовувати для реалізації довільної функції змінних, шо подане в УДНФ. Методика такого синтезу полягає в тому, що задану в УДНФ булеву функцію n - змінних по можливості спочатку мінімізують, приводять до МДІФ. Тоді на n -адресні входи подають вхідні змінні X_i на 2^n інформаційні входи - лог. 1 або лог. 0.

Лог. І подають на такі входи, яким відповідають конституентам одиниці даної функції, а лог. 0 - на решту інформаційних входів мультиплексора. На виході останнього при цьому буде сформована задана функція. Такий самий підхід можне використати і для побудови постійних запам"ятовувальних пристроїв .

Для цього на інформаційні входи подаються фіксовані постійні логічні рівні І або 0, що відповідають двійковому-числу, яке треба запам’ятати. При зчитуванні запам’ятовуваного числа на адресні входи подають код, який відповідає цьому числу.

Важливою перевагою побудови КП на основі мультиплексорів є можливість мінімізації логічної функції. Цьому сприяє застосування методу каскадів або декомпозиції МДНФ заданої логічної функції на підфункції, які, власне, й підлягають спрощенню.

N	X_0	X_1	X_2	Y_1	Y_2
0	0	0	0	0	1
1	0	0	1	1	0
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	1
6	1	1	0	1	0
7	1	1	1	0	0

$Y_1 = k_1 \vee k_3 \vee k_5 \vee k_6$

$Y_2 = k_0 \vee k_2 \vee k_4 \vee k_5$

Розрахуємо параметри компонентів:

$$R = \frac{E - U^1}{nI^1}; \qquad I^1 = 0{,}5 \text{ мА}; \qquad R = \frac{5 - 2{,}4}{4 \cdot 0{,}5 \cdot 10^{-3}} = 1.3 \text{ кОм}.$$

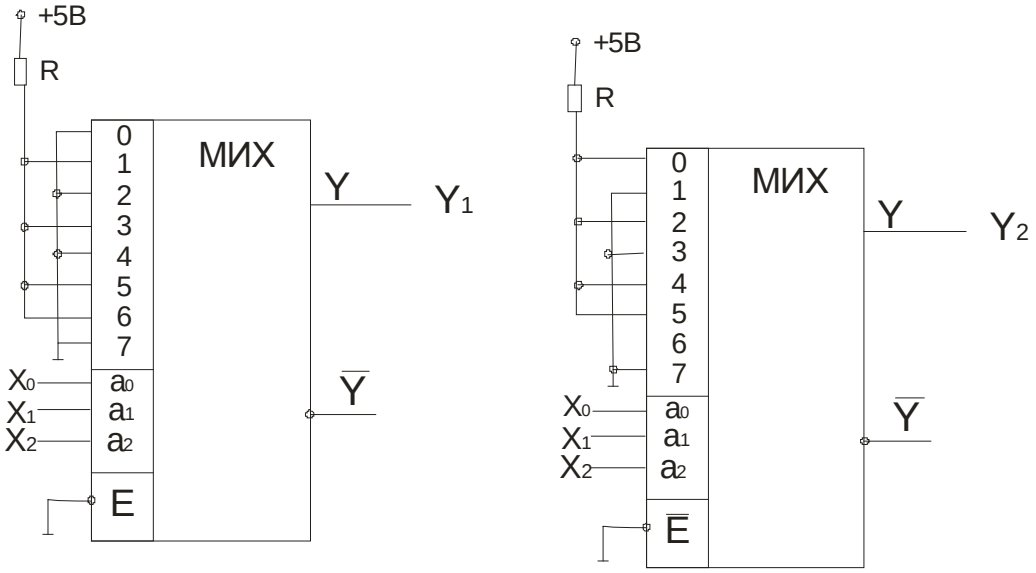


Рис. Схемна реалізація комбінаційного пристрою, заданого таблицею істинності на основі мультиплексора.

Комбінаційні суматори

Це функціональні вузли, що реалізують арифметичне підсумовування (додавання) чисел. Додавання багаторозрядних чисел здійснюється порозрядно з врахуванням переносу у сусідній старший розряд. Тому при побудові суматора необхідно враховувати не лише появу переносу в даному розряді, але й можливість одержання аналогічного переносу від сусіднього молодшого розряду.

За принципом побудови і типом використаних елементів розрізняють комбінаційні та накопичуючі суматори. Результати проміжного порозрядного додавання у накопичуючих суматорах зберігається(запам’ятовується) елементарних комірках пам’яті (запам’ятовувачів),_функцію яких виконують тригери. Комбінаційні суматори" не мають запам’ятовувачів.

У комбінаційних суматорах додавання двійкових чисел здійснюється позиційним паралельним кодом одночасно. На виході результат додавання зникає зразу після припинення дії вхідних сигналів.

Суматори бувають послідовні і паралельні. Вони будуються на основі однорозрядного суматора, що складається з напівсуматора.

Напівсуматор- це пристрій , що має два входи для доданків ***а і в*** і два виходи суми S і переносу *P*, який призначений для виконання арифметичних дій за правилами, що наведені у табл.

1.Напівсуматор виконує елементарне додавання двох однорозрядних двійкових чисел та підсумовування отриманого результату з переносом у старший розряд. З таблиці істинності (табл. 1) ДНФ має вигляд:

$$S = a\bar{b} \vee \bar{a}b = a \oplus b$$

$$P = ab.$$

Табл.1

ab	p	S
0 0	0	0
0 1	0	1
1 0	0	1
1 1	1	0

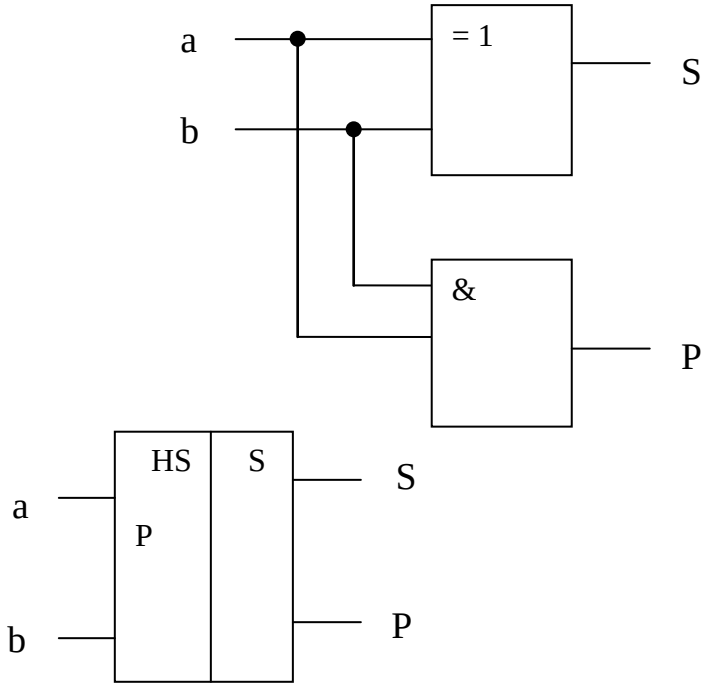


Рис.1 Схема напівсуматора та умовне позначення

Відповідно до наведених виразів логічна структура напівсуматора має містити два НЕ: суматор за модулем 2 і кон'юнктор, що зображені на рис. 1.

Однак у логіці роботи напівсуматора не передбачено переносу з сусіднього молодшого розряду, тому напівсуматор може здійснювати додавання тільки у молодшому розряді двійкових чисел. Поява одиниці переносу при додаванні двох розрядів числа і переносу дещо змінює правило підсумовування двійкових чисел. Такий однорозрядний суматор потребує ще один третій вхід переносу з сусіднього молодшого розряду. Для цього служить так званий повний суматор.

Повний суматор реалізує процедуру додавання двох однорозрядних двійкових чисел з урахуванням переносу з молодшого розряду. Тому він має три входи (a_i,b_i,P_i) і два виходи. (S_i і P_{i+1})**.** Логіка роботи повного суматора наведена у таблиці істинності, де ***(aі,ві)***-доданки двійкових чисел в і-му розряді; ***Pі , Pі+1*** - переноси відповідно з молодшого розряду ***і*** в сусідній старший розряд і+1; ***Sі***- утворена сума в і-му розряді.

Повний суматор має сигнал вхідного P_i та вихідного переносу

a_i	b_i	p_i	P_{i+1}	S_i
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

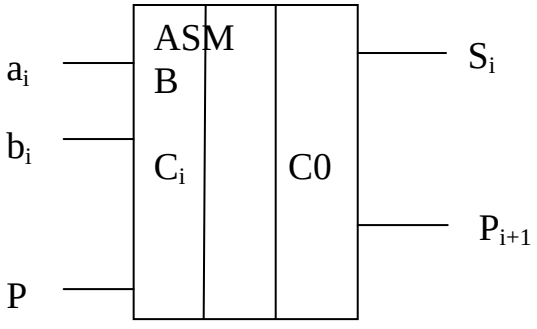


Рис. 2 Умовне позначення повного суматора

За допомогою таблиці будуємо ДНФ:

$$S_i = \bar{a}_i b_i \bar{P}_i \vee a_i \bar{b}_i \bar{P}_i \vee \bar{a}_i \bar{b}_i P_i \vee a_i b_i P_i = \bar{P}_i(\bar{a}_i b_i \vee a_i \bar{b}_i) \vee P_i(\bar{a}_i \bar{b}_i \vee a_i b_i) = \bar{P}_i(a_i \oplus b_i) \vee P_i\overline{(a_i \oplus b_i)} =$$
$$= P_i \oplus a_i \oplus b_i$$

$$P_{i+1} = a_i b_i \bar{P}_i \vee \bar{a}_i b_i P_i \vee a_i \bar{b}_i P_i \vee a_i b_i P_i = a_i b_i \underbrace{(P_i \vee \bar{P}_i)}_{1} \vee P_i(\bar{a}_i b_i \vee a_i \bar{b}_i) = a_i b_i \vee P_i(a_i \oplus b_i)$$

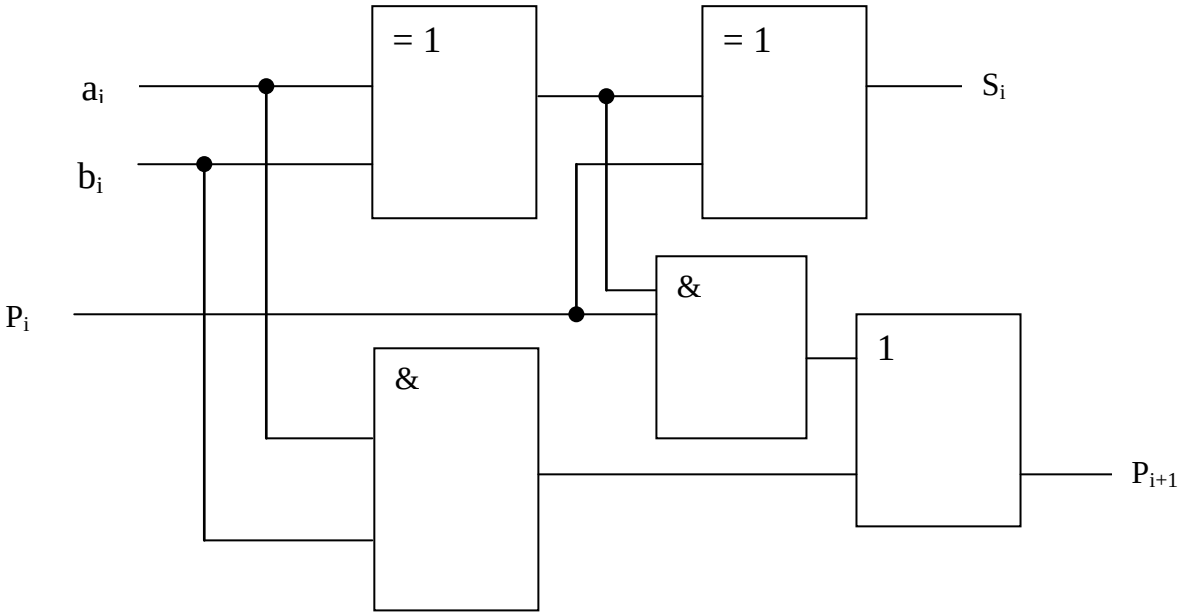


Рис.3 Схема повного суматора однорозрядних чисел.

Для додавання двох n-розрядних чисел A і B необхідно використати n однорозрядних повних суматорів. При цьому підсумовування може бути послідовне і паралельне.

Послідовний суматор додає двійкові числа порозрядно, починаючи з молодшого розряду (рис.4),з допомогою повного суматора. Утворений у даному розряді перенос $\boldsymbol{P_{i+1}}$ з допомогою схеми затримки затримується на 1 такт розряду і подається на вхід $\boldsymbol{P_i}$ суматора у момент надходження наступного розряду доданків.Він забезпечує послідовне додавання чисел розряд за розрядом.

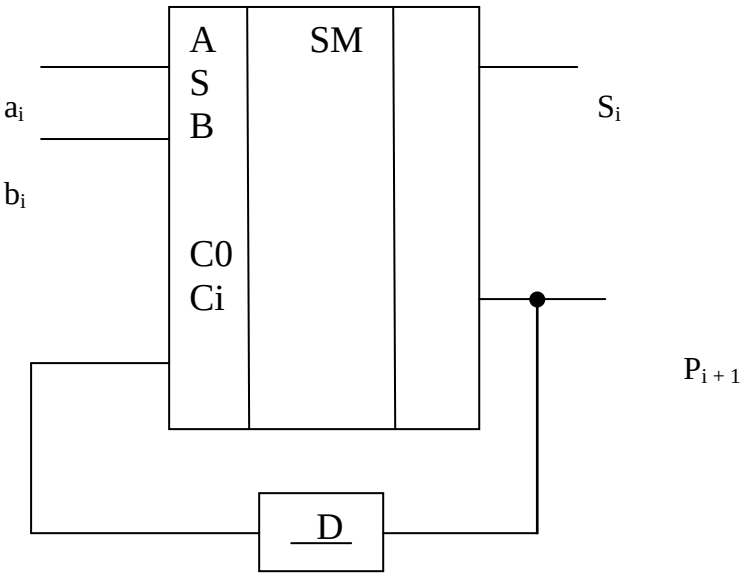


Рис4. Послідовний суматор

Перенос з молодшого розряду враховується при сумуванні наступного старшого розряду числа за рахунок використання схеми затримки D на один такт.

Перевага - простота реалізації. Недолік - низька швидкодія, через необхідність реалізації підсумовування за кількість тактів, що дорівнює числу розрядів чисел.

Паралельний суматор (рис.4) підсумовує два n-розрядних числа одночасно по всіх розрядах і характеризується різними способами передачі переносів від молодших до старших розрядів. Якщо кожен біт переносу здійснюється послідовно з молодшого розряду до старшого, як це показано на рис.4 , то такий паралельний суматор називають суматором з послідовним переносом. На вході переносу молодшого розряду P_0 установлюють сигнали переносів. У процесі послідовного проходження переносу в кожному розряді суматора встановлюється кінцеве значення суми. Очевидно , що протягом цього часу на входах суматора a_i і b_i мають бути присутні сигнали , що відповідають кодам обох доданків. Отже , основна вимога до одно розрядного повного суматора – мінімальна затримка поширення сигналу переносу. Хоч паралельний суматор з послідовним переносом на відміну від послідовного суматора більш швидкодіючий, однак, чим більше число розрядів мають доданки двійкових чисел, тим більша загальна затримка тракту переносу старшого розряду.

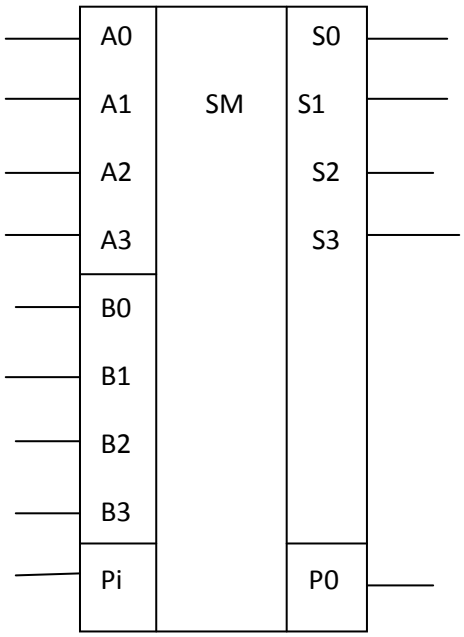
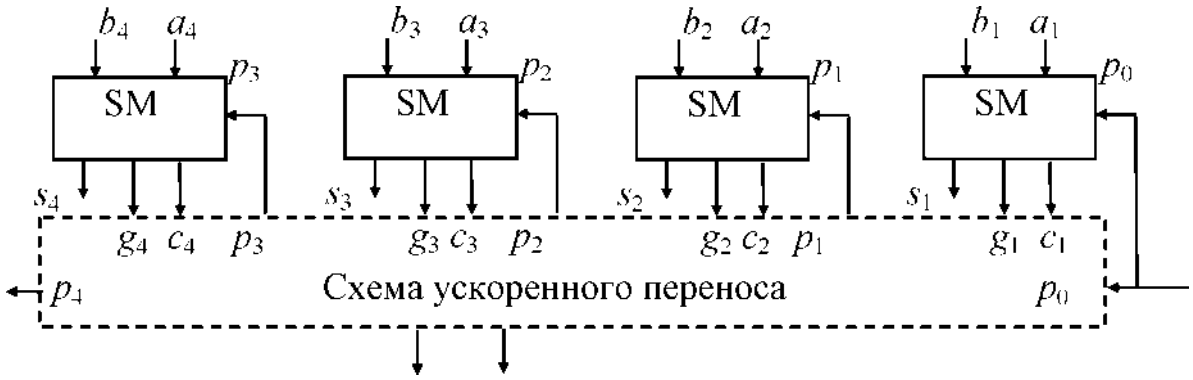


Рис.5. Умовне позначення суматора двох 4- розрядних чисел К155ИМ3

Для зменшення загальної затримки застосовують принцип паралельного переносу, за яким вхідний перенос кожного розряду вибирається незалежно від переносу сусіднього молодшого розряду. Для цього застосовують спеціальні блоки прискореного переносу. Принцип прискореного переносу полягає у тому, що для кожного двійкового і –горозряду додатково утворюються два сигнали – переносу ($g_i = a_i b_i$) і поширення переносу ($h_i = \overline{p_i}\ (a_i + b_i)$). Сигнал g_i виробляється схемою тоді, коли в кожному і-му розряді перенос відбувається внаслідок комбінації доданків a_i і b_i , а сигнал h_i показує, передається отриманий у молодшому розряді сигнал переносу P_i далі чи ні. Тому g_i ще називають функцією генерації переносу, а P_i – функцією поширe



Цифрові компаратори

Це арифметичні пристрої, що призначені для порівняння двох чисел, що подані у двійковому (двійково-десятковому) коді.

Найпростіший компаратор виявляє лише факт рівності або нерівності двох поданих на його входи n- розрядних чисел (операндів) A і B, і формує на виході однобітовий сигнал рівності (1) або нерівності (0) цих чисел.

Рівність, зокрема, двох операндів a і b визначається логічною операцією однозначності, або еквівалентності

$$\mathbf{y} = \begin{cases} \mathbf{1 \text{ при } a = b} \\ \mathbf{0 \text{ при } a \neq b} \end{cases}, \quad (1)$$

тобто логічною операцією ВИЙНЯТКОВО АБО-НЕ, яка реалізується суматором-інвертором за mod2.

Покажемо це:

Операція рівнозначності (1) записується як

$$Y = ab \vee \bar{a}\bar{b}.$$

a	b	y
0	0	1
0	1	0
1	0	0
1	1	1

Звідси

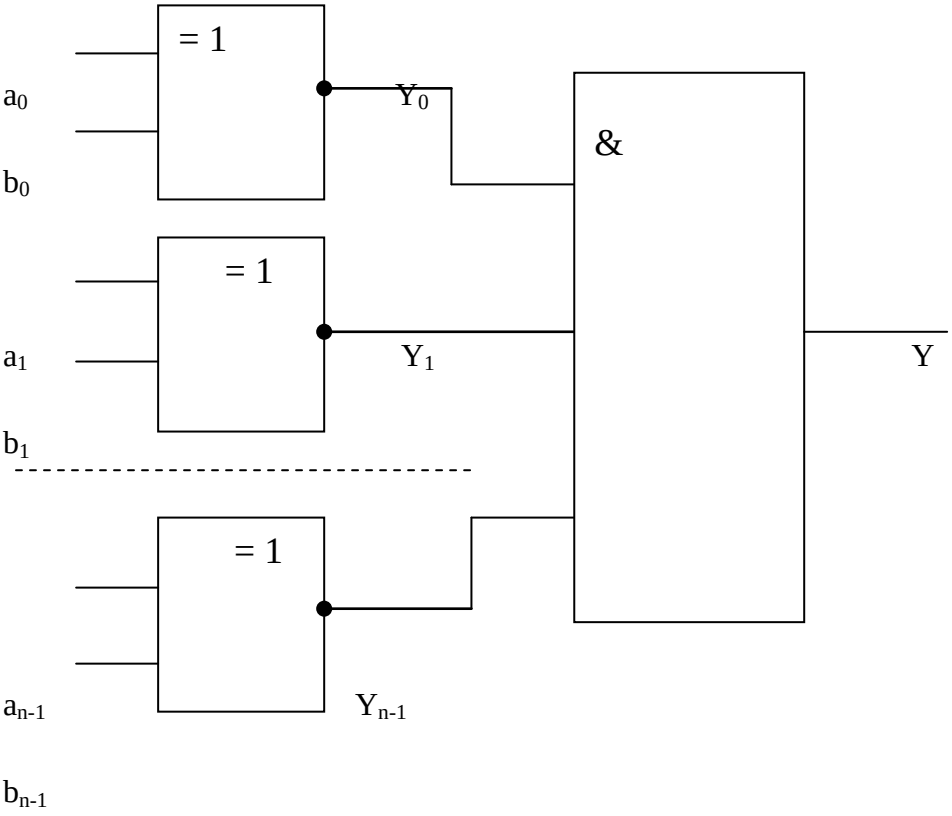
$$Y = ab \vee \bar{a}\bar{b} = \overline{\overline{ab \vee \bar{a}\bar{b}}} = \overline{\overline{ab}\,\overline{\bar{a}\bar{b}}} = \overline{(\bar{a} \vee b) \cdot (a \vee \bar{b})} = \overline{a\bar{a} \vee \bar{a}b \vee a\bar{b} \vee b\bar{b}} = \overline{\bar{a}b \vee a\bar{b}} = \quad = a \oplus b.$$

Реалізація його доволі просто здійснюється у довільному базисі.

Порозрядну рівність n- розрядних операндів найпростіше реалізувати за допомогою суматорів-інверторів за mod2 і кон'юнкторів. Такий компаратор рівностей порівнює окремі розряди n-розрядних чисел за формулою:

$$Y^{(n)} = y_0 y_1 \dots y_{n-1} = \overline{(a_0 \oplus b_0)}\,\overline{(a_1 \oplus b_1)} \dots \overline{(a_{n-1} \oplus b_{n-1})}$$

Реалізація цієї функції:



Аналогічний результат отримаємо і при синтезі компаратора у базисі суматорів за mod2 і диз'юнкторів. Якщо застосувати закон дуальності, то вихідна функція такого компаратора описується виразом:

$$Y = \overline{\overline{y_0} \vee \overline{y_1} \vee \dots \vee \overline{y_{n-1}}} = \overline{(a_0 \oplus b_0) \vee (a_1 \oplus b_1) \vee \dots \vee (a_{n-1} \oplus b_{n-1})}$$

Компаратори порівняння n-розрядних чисел можна будувати за двома принципами:

1. Логічним
2. Арифметичним.

Логічний принцип базується на синтезі за таблицею істинності.

Порівняння n-розрядних операндів A і B є більш складною процедурою - воно визначається системою нерівностей, що складається з двох функцій:

$$Y_A^{(n)} = \begin{cases} 1 \text{ при } A < B \\ 0 \text{ при } A \geq B \end{cases}; \qquad Y_B^{(n)} = \begin{cases} 1 \text{ при } A > B \\ 0 \text{ при } A \leq B \end{cases}.$$

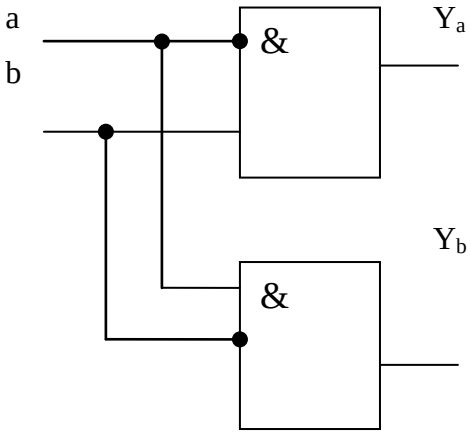
Таблиця істинності компаратора двох однорозрядних чисел має вигляд:

a b	$Y_a Y_b$
0 0	0 0
0 1	1 0
1 0	0 1
1 1	0 0

Звідси: $Y_A = \bar{a}b$ – заборона по a

$Y_B = a\bar{b}$ – заборона по b.

Схема приведена на рис 5.



При синтезі n-розрядного компаратора порівняння операторів A і B користуються таблицею істинності з подальшою мінімізацією функцій порівняння A і B.

Арифметичний принцип порівняння двійкових n-розрядних чисел A і B можна здійснювати за допомогою суматора шляхом реалізації операції віднімання A-B і схеми аналізу отриманого результату.

Можливі 3 ситуації, які повинен зафіксувати компаратор

A>B; A=B і A<B

Відомо що операцію віднімання двох чисел можна замінити додаванням якщо код від'ємника перетворити наприклад у доповняльний код, який утворюється з оберненого шляхом додавання одиниці до молодшого розряду.

Аналогічно можна виконати функції порівняння за допомогою суматора.

Для цього:

1.код числа A подається на суматор у прямій формі;

2.код числа B подається на суматор у інверсній формі;

3.на вхід C1 переносу P_i подається одиниця.

Тоді сигнали виходу суми Si і переносу Pi+1 визначатимуть результат порівняння. Зокрема якщо:

$P_{i+1} = 1$ і $S_i \neq 0$ означає, що A>B

$P_{i+1} = 1$ і $S_i = 0$ означає, що A=B

$P_{i+1} = 0$ означає, що A<B.

Цей алгоритм справедливий для додатних чисел A і B без врахування їх знаків.

Проілюструємо порівняння двох n-розрядних чисел A і B на прикладах що охоплюють три випадки.

1 випадок: A > B, нехай A=14; B=11

$A_2 = 1110$;

$B_2 = 1011 \rightarrow \widetilde{B_2} = 0101;$ $\quad + \begin{array}{r} 0.1110 \\ 0.0101 \\ \underbrace{1.0011} \end{array}$

Ознакою A > B є $P_{i+1} = 1; S_i \neq 0.$

2 випадок A=B; A=B=11;

$A_2 = B_2 = 1011 \ \widetilde{B_2} = 0101;$ $\quad + \begin{array}{r} 0.1011 \\ 0.0101 \\ \underbrace{1.0000} \end{array}$

Ознакою A=B є S=0; $P_{i+1} = 1$

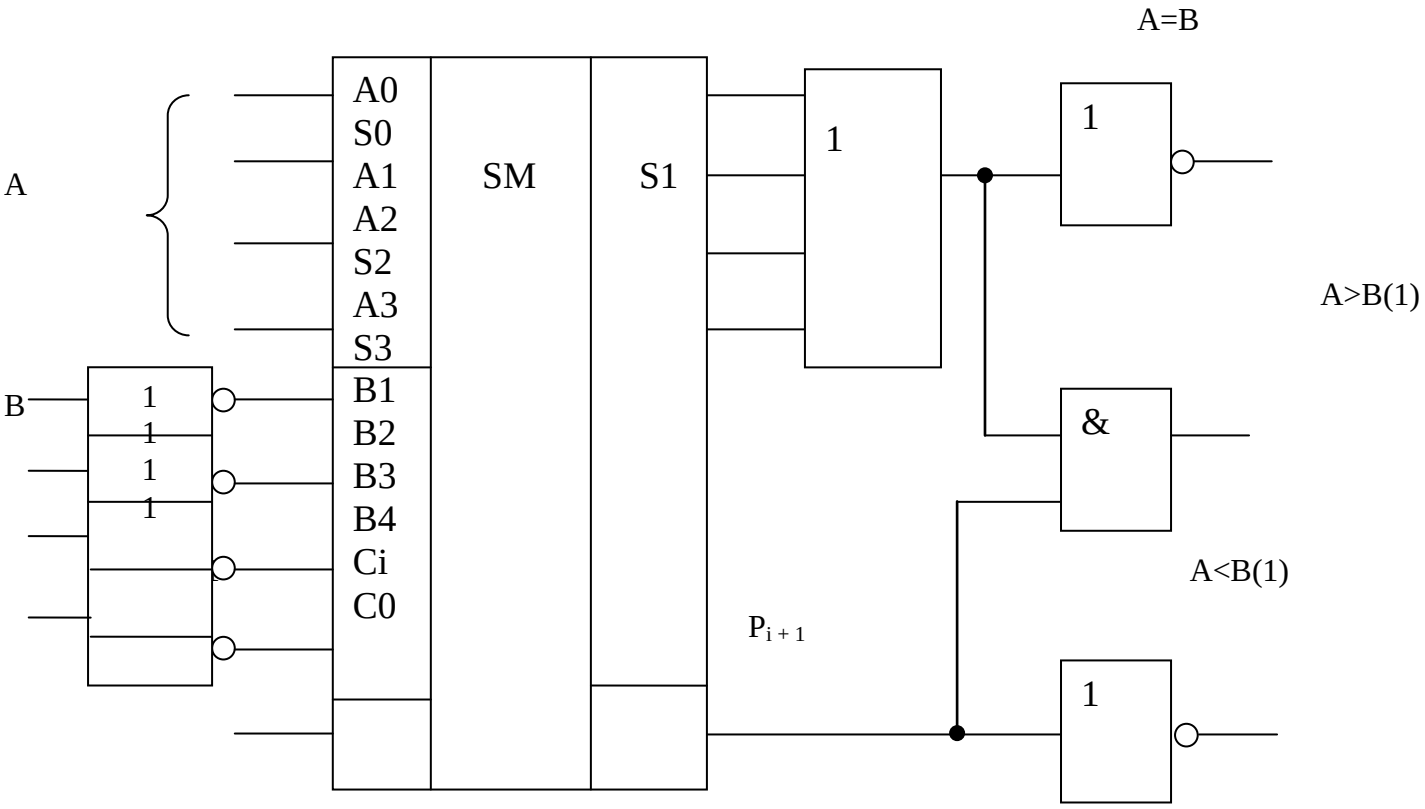
3 випадок A<B; A=11; B=14;

$A_2 = 1011;$

$B_2 = 1110 \rightarrow \widetilde{B_2} = 0010;$ $\quad + \begin{array}{r} 0.1011 \\ 0.0010 \\ \underbrace{0.1101} \end{array}$

Ознакою A=B є $P_{i+1} = 0.$

Схема чотирирозрядного компаратора, що реалізує арифметичний принцип порівняння.



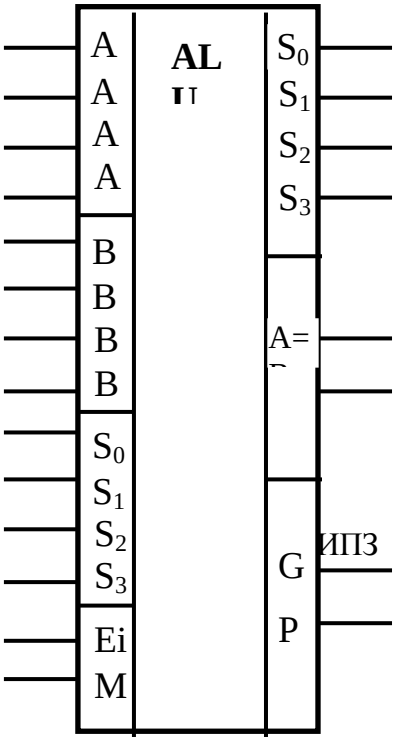
Промисловістю випускаються цифрові багаторозрядні компаратори як окремі вироби. 4-розрядрні ТТЛШ-555СП1; 531СП1.

Вони мають додаткові інвертори для операнда В і додаткові три входи аналізу I>(A>B); I=(A=B) і I<(A<B). Останні призначені для утворення схеми нарощування розрядності операндів, тобто для каскадування послідовним або пірамідальним способом під'єднання однотипних мікросхем.

Арифметико – логічні пристрої (АЛП)

Для виконання над операндеми А і В як арифметичних, так і логічних операцій доцільно застосовувати мікросхеми універсальних арифметико-логічних пристроїв /АЛЛ/, які можна знайти у складі серій ТТЛ. Найчастіше вони мають чотирирозрядні входи операндів А і В і придатні для нарощування їх розрядності. Для виконання тих чи інших логічних або арифметичних операцій служать спеціальні керуючі входи. Подаючи на них чотирирозрядний код і задавши тип операції (арифметичної чи логічної), можна замовити мікросхемі одну з 16 арифметичних або одну з 16 логічних операцій. Через широкі функціональні можливос¬ті мікросхеми АЛЛ входять до складу мікропроцесорів ЕОМ. У комплекті з АЛЛ випускають ще мікросхему, яка являє собою тракт групового переносу для виконання прискореного підсумовування багаторозрядних чисел.

До мікросхем АЛЛ належать із серій ТТЛ КІ55ИПЗ /із схемою прискореного переносу КІ55ИП4/, із серій КМОН 564ШЗ /відповідно 564ИП4/.



АЛП має чотири групи входів:A_0 – A_3 та B_0 – B_3 для двох чисел, з якими потрібно зробити операції. Також є 4 виходи Y_0 – Y_3, на яких фіксується результат арифметичної операції при(M = 1 або M = 0). Потрібну операцію вибирають за допомогою коду на вході S_0 – S_3.

Хід роботи:

1.Реалізувати схему з використанням КП для перетворення чотирирозрядного двійкового коду в код Грея.

2. Реалізувати схему з використанням КП для перетворення коду Грея в чотирирозрядний двійкового код

3.Реалізувати за допомогою КП схему для перетворення чотирирозрядного двійкового коду в доповняльний код.

4. Реалізувати за допомогою КП схему для перетворення доповняльного коду в чотирирозрядний двійковий код .

5. Реалізувати логічну функцію за допомогою мультиплексора, розрахувати значення опору R:

1) Y = $\sum(1,3,4,7,)$
2) Y = $\sum(1,2,3,6,7,)$
3) Y = $\prod(1,3,4,6)$ (за допомогою MUX 8 → 1)
4) Y = $\sum(1,3,4,7,)$
5) Y = $\prod(1,3,4,6,7)$ (за допомогою MUX 8 → 1)
6) Y = $\sum(1,3,4,7,13)$
7) Y = $\sum(1,2,4,7,14,15)$
8) Y = $\sum(1,2,4,5,7,13,15)$
9) Y = $\sum(1,5,6,7,)$
10) Y = $\sum(0,3,4,7,12)$
11) Y = $\sum(1,2,3,4,7,11)$
12) Y = $\sum(1,3,4,7,10)$

6. За допомогою дешифратора реалізувати логічну функцію.

1)У=$\sum(0,1,5,6,9,14)$
2) У=$\sum(0,2,5,7,8,11)$
3) У=$\sum(0,3,6,7,8,11)$
4) У=$\sum(3,4,9,15)$
5) У=$\sum(1,4,6,10,13)$
6) У=$\sum(1,5,6,8,14)$
7) У=$\sum(0,3,9,11,12,15)$
8) У=$\sum(4,5,9,10,12)$
9) У=$\sum(0,2,4,6,8,10,12,14)$
10) У=$\sum(1,3,5,7,9,11,13,15)$
11) У=$\sum(0,4,5,7,8,14)$

12) У=$\sum(0,1,3,7,9,11)$

7.За допомогою дешифратора з інверсними виходами побудувати схему демультиплексора 1-10.

8.Побудувати схему послідовного суматора двох чотирирозрядних чисел на базі повних однорозрядних суматорів.

9.Реалізуйте шифратор, що перетворює цифри від 0 до 9 в двійковий-десятковий код.

10*.**Побудувати перемножувач двох чотирирозрядних двійкових чисел на основі суматорів.

Контрольні запитання:

Де застосовуються шифратори та дешифратори? За яким принципом вони будуються?

2. У чому відмінність між мультиплексором та демультиплексором?
3. Що таке каскадування цифрових пристроїв і як воно здійснюється?
4. Що таке комбінаційний суматор і за яким принципом його синтезують?
5. У чому відмінність між напівсуматором та повним суматором?

Де їх застосовують?

6. Які функції може виконувати арифметично-логічний пристрій?

Лабораторна робота №5

Дослідження тригерів

1 Мета роботи:

Метою роботи є експериментальне дослідження роботи різних типів тригерів.

2 Короткі теоретичних відомості

Тригери призначені для запам'ятовування двійкової інформації. Використання тригерів дозволяє реалізовувати пристрої оперативної пам'яті (тобто пам'яті, інформація в якій зберігається тільки на час обчислень). Проте тригери можуть використовуватися і для побудови деяких цифрових пристроїв з пам'яттю, таких як лічильники, перетворювачі послідовної коди в паралельний або цифрові лінії затримки.

2.1 RS-тригер

Основним тригером, на якому базується решта всіх тригерів є RS-тригер.

RS-тригер має два логічні входи:

- *R - установка 0 (від слова reset);*
- *S - установка 1 (від слова set).*

RS-тригер має два виходи:

- *Q - прямий;*
- *Q- зворотній (інверсний).*

Стан тригера визначається станом прямого виходу. Простий RS-тригер складається з двох логічних елементів, охоплених перехресним позитивним зворотним зв'язком (рис 2.1).

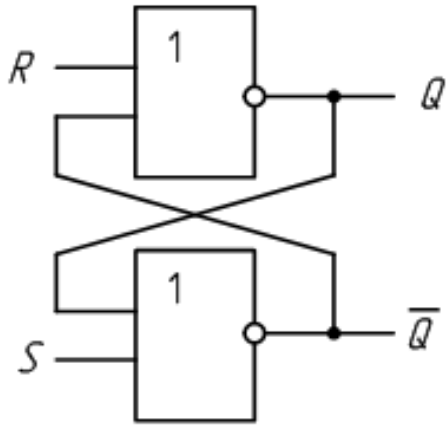


Рис 2.1 - Схема простого RS-тригера

Розглянемо роботу тригера:

Нехай R=0, S=1. Нижній логічний елемент виконує логічну функцію АБО-НЕ, тобто 1 на будь-якому його вході приводить до того, що на його виході буде логічний нуль Q=0. На виході Q буде 1 (Q=1), оскільки на обидва входи верхнього елементу

подані нулі (один нуль - з входу R, інший - з виходу). Тригер знаходиться в одиничному стані. Якщо тепер прибрати сигнал установки (R=0, S=0), на виході ситуація не зміниться, оскільки не дивлячись на те, що на нижній вхід нижнього логічного елементу поступатиме 0, на його верхній вхід поступає 1 з виходу верхнього логічного елементу. Тригер знаходитиметься в одиничному стані, поки на вхід R не поступить сигнал скидання.

Нехай тепер R=1, S=0. Тоді Q=0, а =1. Тригер перемкнувся в "0". Якщо після цього прибрати сигнал скидання (R=0, S=0), то все одно тригер не змінить свого стану.

Для опису роботи тригера використовують таблицю станів (переходів). Позначимо:

- *Q(t) - стан тригера до надходження сигналів, що управляють (зміни на входах R і S);*
- *Q(t+1) - стан тригера після зміни на входах R і S.*

Таблиця 2.1 - Таблиця переходів RS тригера в базисі АБО-НЕ

R	S	Q(t)	Q(t+1)	Пояснення
0	0	0	0	Режим зберігання інформації R=S=0
0	0	1	1	
0	1	0	1	Режим встановлення одиниці S=1
0	1	1	1	
1	0	0	0	Режим встановлення нуля R=1
1	0	1	0	
1	1	0	*	R=S=1 заборонена комбінація
1	1	1	*	

RS-тригер можна побудувати і на елементах "І-НЕ" (рис 2.2).

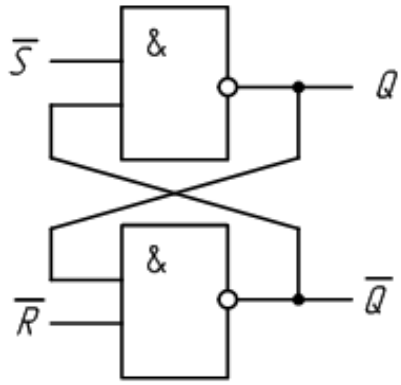


Рис. 2.2 - Схема RS-тригера, побудованого на схемах "АБО-НЕ"

Входи R і S інверсні (активний рівень "0"). Перехід (перемикання) цього тригера з одного стану в інше відбувається при установці на одному з входів "0". Комбінація R=S=0 є забороненою.

Таблиця 2.2 - Таблиця переходів RS тригера в базисі "АБО-НЕ"

R	S	Q(t)	Q(t+1)	Пояснення
0	0	0	*	R=S=0 заборонена комбінація
0	0	1	*	
0	1	0	0	Режим установки нуля R=0
0	1	1	0	
1	0	0	1	Режим установки одиниці S=0
1	0	1	1	
1	1	0	0	Режим зберігання інформації R=S=0
1	1	1	1	

2.2 Синхронний RS-тригер

Схема RS-тригера дозволяє запам'ятовувати стан логічної схеми, але оскільки при зміні вхідних сигналів може виникати перехідний процес, то запам'ятовувати стани логічної схеми потрібно тільки в певні моменти часу, коли всі перехідні процеси закінчені, і сигнал на виході комбінаційної схеми відповідає виконуваній нею функції. Це означає, що більшість цифрових схем вимагають сигналу синхронізації (тактового сигналу). Всі перехідні процеси в комбінаційній логічній схемі повинні закінчитися за час періоду синхросигналу, що подається на входи тригерів. Тригери, що запам'ятовують вхідні сигнали тільки у момент часу, визначуваний сигналом синхронізації, називаються синхронними. Принципова схема синхронного тригера RS приведена на рис. 2.3.

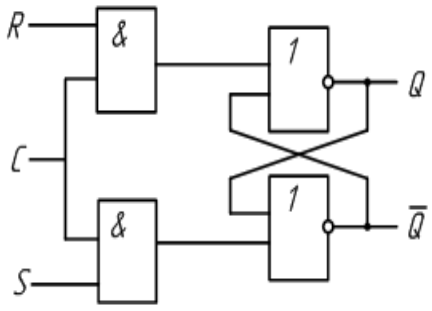


Рис. 2.3 - Схема синхронного RS-тригера

Таблиця 2.3 - Таблиця переходів синхронного RS-тригера

R	S	C	Q(t)	Q(t+1)	Пояснення
0	0	1	0	0	Режим зберігання інформації R = S = 0
0	0	1	1	1	
0	1	1	0	1	Режим установки одиниці S =1
0	1	1	1	1	
1	0	1	0	0	Режим установки нуля R=1
1	0	1	1	0	

1	1	1	0	*	R = S = 1 заборонена комбінація
1	1	1	1	*	

У таблиці 2.3. під сигналом С мається на увазі синхроімпульс. Без синхроімпульса синхронний тригер RS зберігає свій стан.

2.3 D - тригер

D-тригер має 1 інформаційний вхід (D-вхід). Бувають тільки синхронні D-тригери. Стан інформаційного входу передається на вихід під дією синхроімпульса (вхід С).

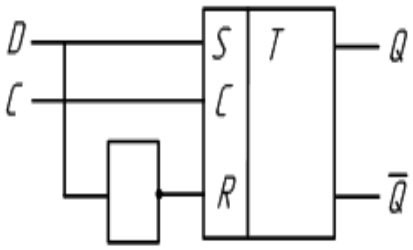


Рис. 2.4 - Схема D-тригера на основі синхронного RS-тригера

Таблиця 2.4 - Таблиця переходів D-тригера

C	D	Q(t)	Q(t+1)	Пояснення
0	*	0	0	Режим зберігання інформації
0	*	1	1	
1	0	*	0	Режим запису інформації
1	1	*	1	

Якщо на вході D - "1", то по приходу синхроімпульса Q = 1. Якщо на D "0", то Q =0.

2.4 Лічильний тригер (Т-тригер)

Т-тригер має один лічильний інформаційний вхід. Тригер перемикається кожного разу в протилежний стан, коли на вхід Т поступає керуючий сигнал.

Таблиця 2.5 - Таблиця переходів Т тригера

T	Q(t)	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

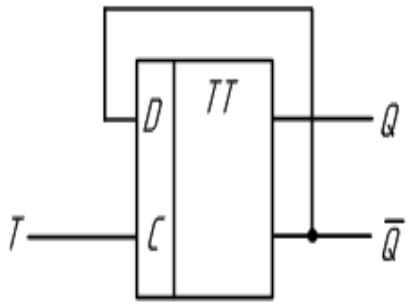


Рис. 2.5 - Схема Т-тригера на основі двоступінчатого D-тригера

2.5 Універсальний тригер (JK-тригер)

Такий тригер має інформаційні входи J і K, які по своєму впливу аналогічні входам S і R тактованого RS-тригера:

- *при J=1, K=0 тригер по тактовому імпульсу встановлюється в стан Q=1;*
- *при J= 0, K=1 - перемикається в стан Q=0;*
- *при J=K=0 - зберігає раніше прийняту інформацію.*

Але на відміну від синхронного RS-тригера одночасна присутність логічних 1 на інформаційних входах не є для JK-тригера забороненою комбінацією і приводить тригер в протилежний стан.

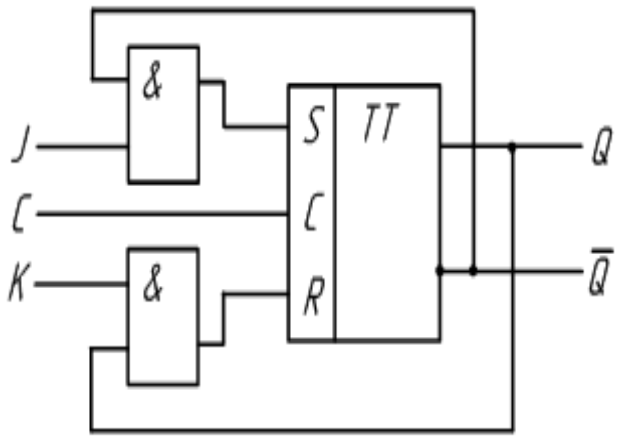


Рис. 2.6 - Схема JK-тригера на основі двоступінчатого синхронного RS-тригера.

Таблиця 2.6 - Таблиця переходів JK тригера

K	J	C	Q(t)	Q(t+1)
0	0	1	0	0
0	0	1	1	1
0	1	1	0	1
0	1	1	1	1

1	0	1	0	0
1	0	1	1	0
1	1	1	0	1
1	1	1	1	0

3 Зміст звіту

3.1 Мета роботи.
3.2 Привести умовні графічні позначення досліджуваних тригерів
3.3. Таблиця переходів досліджуваних тригерів.
3.4. Скласти схеми на лабораторному стенді згідно із завданням
3.5 Висновки за кожним завданням.

4 Контрольні запитання

5.1 Чим визначається швидкодія тригера?
5.2 Накреслити схему RS-тригера на логічних елементах "АБО-НЕ" і пояснити принцип його роботи.
5.3 Чому JK-тригер називається універсальним?
5.4 Пояснити по таблиці переходів роботу D-тригера.
5.6 Способи опису послідовних цифрових пристроїв.

Лабораторна робота №6

Дослідження регістрів

1 Мета роботи: Метою роботи є вивчення принципу роботи схем тригерних регістрів і отримання практичних навичок у виконанні мікроопераций на регістрах в статичному режимі.

2 Короткі теоретичні відомості

Регістри призначені для зберігання і перетворення багаторозрядних двійкових чисел. Для запам'ятовування окремих розрядів числа можуть застосовуватися тригери різних типів. Поодинокий тригер можна вважати однорозрядним регістром.

Занесення інформації в регістр називається операцією запису. Операція видачі інформації з регістра - зчитування.

Перед записом інформації в регістр, його необхідно обнулити.

Класифікація регістрів :

1 за способом введення/виведення інформації :

- паралельні (регістри зберігання) - інформація вводиться і виводиться одночасно по усіх розрядах;
- послідовні (регістри зсуву) - інформація біт за бітом «проштовхується» через регістр і виводиться також послідовно;
- комбіновані - паралельне введення і послідовний вивід (і навпаки).

2 за способом представлення інформації :

- однофазні - інформація представляється в прямому або зворотному (інверсному) вигляді;
- парафазні - інформація представляється і в прямому, і в зворотному вигляді.

2.1 Паралельний регістр

Паралельні регістри здійснюють прийом і видачу інформації в паралельному коді, а це означає, що для передачі кожного розряду використовується окрема лінія.

Для запису інформації в регістр на його вхідних виводах (**D0 — D3) треба встановити логічні рівні, після чого на вхід синхронізації (C) подати синхронізуючий імпульс — логічну одиницю. Після цього на виходах Q0 — Q3** з'явиться записане слово. Регістри запам'ятовують вхідні сигнали тільки у момент часу, визначений сигналом синхронізації.

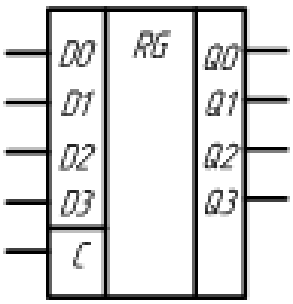


Рис. 2.1 - Умовно-графічне позначення паралельного регістра

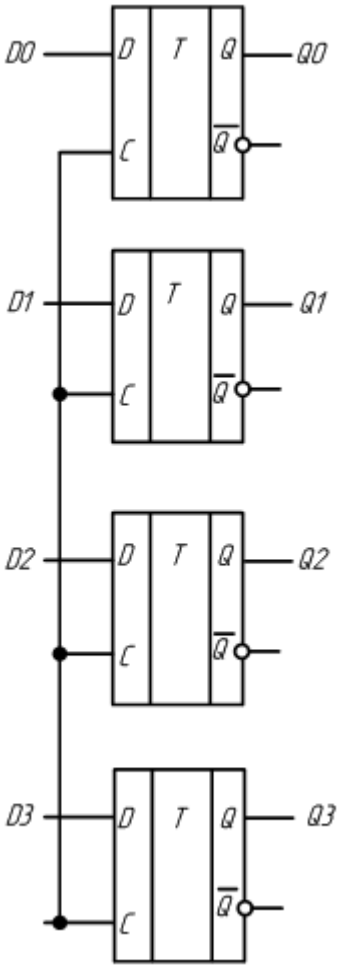


Рис. 2.2 - Схема паралельного регістра

2.1 Послідовні регістри

Окрім паралельного з'єднання тригерів для побудови регістрів використовуються послідовне з'єднання цих елементів.

Послідовний регістр (регістр зсуву) зазвичай служить для перетворення послідовного коду в паралельний і навпаки. Застосування послідовного коду пов'язане з необхідністю передачі великої кількості двійкової інформації по обмеженій кількості з’єднувальних ліній. Якщо двійкові розряди послідовно біт за бітом передавати по одному провіднику, то можна значно скоротити розміри сполучних ліній на платі (і розміри корпусів мікросхем).

Принципова схема послідовного регістра, зібраного на основі D -триггеров і що дозволяє здійснити перетворення послідовного коду в паралельний, приведена на рис. 2.3.

Розглянемо роботу цього регістра.

Можна припустити, що на початку усі тригери регістра знаходяться в стані логічного нуля, тобто Q0=0, Q1=0, Q2=0, Q3=0. Якщо на вході D-тригера Т1 має місце логічний 0, то подавання синхроімпульсів на входи «С», тригерів не міняє їх стану.

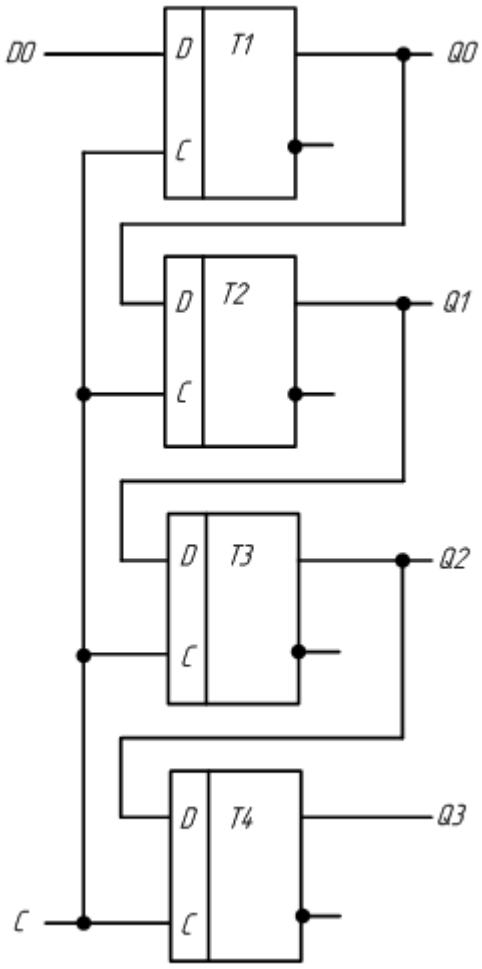


Рис. 2.3 - Схема послідовного регістра

Як випливає з рис 2.3, синхроімпульси поступають на відповідні входи усіх тригерів регістру одночасно і записують в них те, що має місце на їх інформаційних входах. На інформаційних входах тригерів Т2, Т3, Т4 - рівні логічного «0», оскільки інформаційні входи подальших тригерів сполучені з виходами попередніх тригерів, що знаходяться в стані логічного «0», а на вхід «D» першого тригера, по умові прикладу, подається «0» із зовнішнього джерела інформації. При поданні на вхід «D» першого тригера «1», з приходом першого синхроімпульса, в цей тригер запишеться «1», а в інші тригери - «0», оскільки до моменту поступлення фронту синхроімпульса на виході тригера Т1 ще був присутнім логічний «0». Таким чином, в тригер Т1 записується та інформація (той біт), яка була на його вході «D у момент поступлення фронту синхроімпульса і так далі

При поступленні другого синхроімпульсу логічна «1» з виходу першого тригера, запишеться в другий тригер, і в результаті відбувається зсув спочатку записаної «1» з тригера Т1 в тригер Т2, з тригера Т2 в тригер Т3 і так далі. Таким чином, робиться послідовний зсув інформації (у послідовному коді), що поступає на вхід регістра, на один розряд вправо в кожному такті синхроімпульсів.

Після вступу чотирьох синхроімпульсів регістр виявляється повністю заповненим розрядами числа, що поступає через послідовний вхід «D». Впродовж наступних чотирьох синхроімпульсів робиться послідовний порозрядний вивід з регістра записаного числа, після чого регістр стає повністю очищеним (регістр виявиться повністю очищеним тільки за умови подання на його вхід рівня «0» в режимі виведення записаного числа).

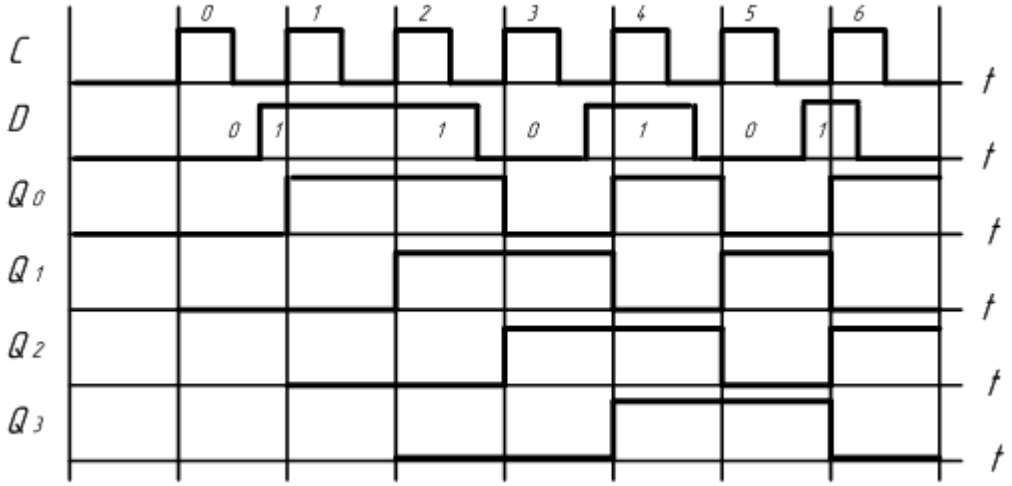


Рис. 2.4 - Часові діаграми, що пояснюють роботу регістра зсуву

Лабораторна робота №7

Дослідження лічильників

1 Мета роботи

Дослідити принципи побудов двійкових лічильників різних типів

2 Короткі теоретичні відомості

Лічильник – пристрій для підрахунку кількості вхідних імпульсів.

Параметри лічильника:

- модуль лічби М – кількість стабільних станів;
- Ємність Е – максимальное число, яке може бути записано в лічильник (Е=М-1);
- Швидкодія (швидкість переходу із стану «всі 1» в стан «всі 0» і навпаки).

Класифікація:

1 По напрямку лічби:

- додавальні;
- віднімальні;
- реверсивні;

2 По способу побудови ланцюга переносу:

- з послідовним переносом;
- з паралельним переносом;
- з комбінованим переносом;

3 По способу переключення тригера:

- синхронні;
- асинхронні.

2.1 Простий додавальний асинхронний лічильник

Лічильник являє собою декілька послідовно з'єднаних тригерів. По кожному вхідному імпульсу лічильний тригер змінює свій стан на протилежний.

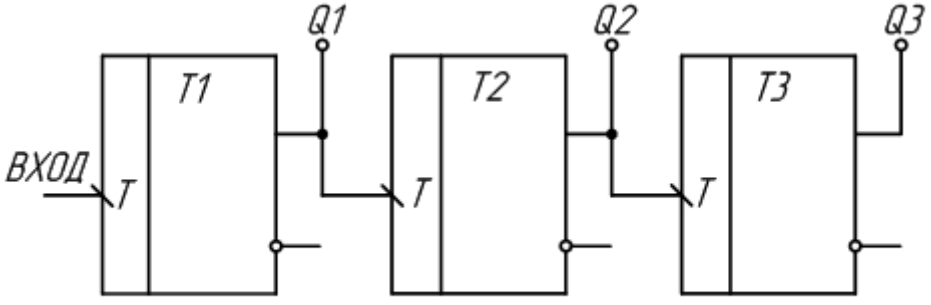


Рисунок 2.1 – Простий додавальний асинхронний лічильник

Якщо синхронізуючий вхід відзначений як «\», то перемикання тригера відбувається по зрізу, а якщо «/» - то по фронту.

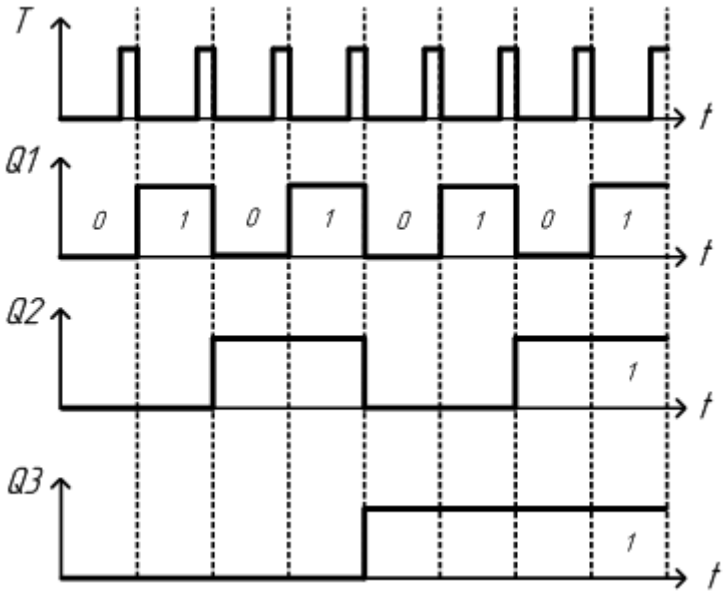


Рисунок 2.2 – Часова діаграма роботи додавального асинхронного лічильника

Для того щоб розібратись, як працює схема двійкового лічильника, скористаємось часовими діаграмами сигналів на вході і виходах даної схеми, приведеними на рисунку 2.2.

Нехай початковий стан всіх тригерів лічильника буде нульовим. Цей стан ми бачимо на часових діаграмах. Запишемо його в таблицю 2.1. Після поступлення на вхід лічильника тактового імпульсу (який сприйматься по зрізу), перший тригер змінює свій стан на протилежний, тобто одиницю.

Запишемо новий стан виходів лічильника в ту ж таблицю. Оскільки після приходу першого імпульсу змінився стан першого тригера, то цей тригер містить молодший розряд двійкового числа (одиницю).

Таблиця 2.1 – Зміна рівнів на виході додавального двійкового лічильника при поступленні на його вхід імпульсів.

Номер вхідного імпульсу	**Q2**	**Q1**	**Q0**
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0

Подамо на вхід лічильника, ще один тактовий імпульс. Значення першого тригера знову зміниться на протилежне. На цей раз на виході першого тригера,а значить і на вході другого утвориться зріз. Це означає, що другий тригер теж змінить свій стна на протилежний. Це чітко видно

на часових діаграмах приведених на рисунку 2.2. Запишемо новий стан виходів лічильника в таблицю 2.1. В цьому рядку таблиці утворилось двійкове число 2. Воно співпадає з номером вхідного імпульсу. Продовжуючи аналіз часової діаграми, можна визначити, що на виходах приведеної схеми лічильника послідовно з'являються цифри від 0 до 7. Ці цифри записані в двійковій формі. При поступленні на лічильний вхід лічильника чергового імпульсу, вміст його тригерів збільшується на 1. Тому такі лічильники отримали назвудодавальних двійкових лічильників. Якщо інформацію знімати з обернених виходів тригерів, то отримаємо піднімальний лічильник.

2.2 Простий віднімальний асинхронний лічильник

Розглянемо схему лічильника на тригерах, які перемикаються по фронту вхідних імпульсів, рис. 2..3

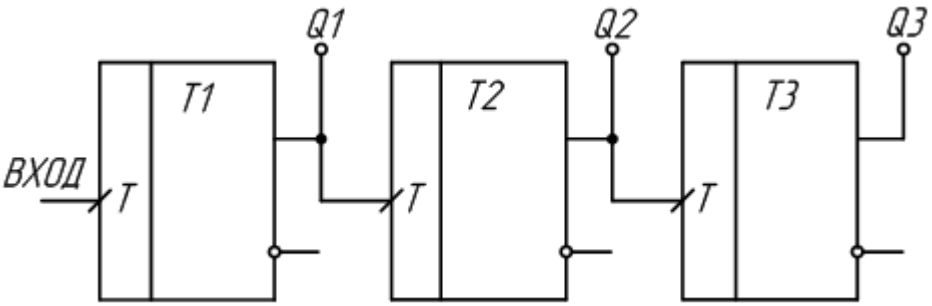


Рисунок 2.3 – Віднімальний лічильник

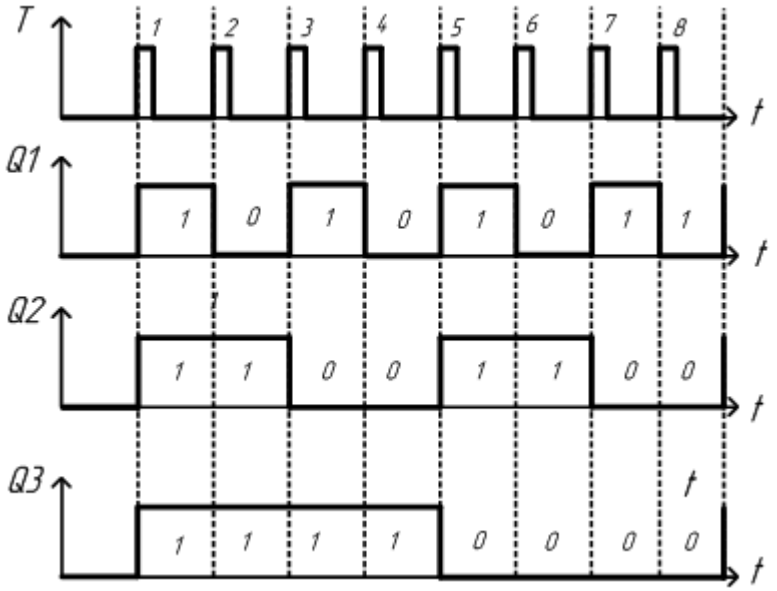


Рисунок 2.4 – Часова діаграма

З часової діаграми бачимо, що ми отримали піднімальний лічильник. Якщо інформацію знімати з інверсного виходу, то вийде додавальний лічильник.

2.3 Лічильник з довільним модулем лічби

Для побудови такого лічильника можна використати двійковий лічильник модуль лічби якого більший від необхідного нам значення.

Нехай потрібно синтезувати лічильник з модулем лічби M=10. В 4-х розрядного лічильника модуль лічби дорівнює 16 (більше 10). Схема лічильника представляє собою 4 послідовно з'єднаних лічильних тригера, в яких є скидаючий вхід R.

Число 10 в двійковій системі числення представляется - 1010. Коли на виходах лічильника буде код 1010, на виході елемента «І» з'явиться логічна одиниця, яка запустить схему скидання. Тривалість імпульсу на виході системи гашення повинна бути достатньою для гарантованого скидання всіх тригерів лічильника в 0. Розряди числа 1010, які дорівнюють 1,подаються на схему «І» з прямих виходів тригерів, а рівні 0 – з інверсних. Таким чином, як тільки лічильник дорахує до 10, відбудеться обнулення тригерів і лічба поновиться з коду 0000.

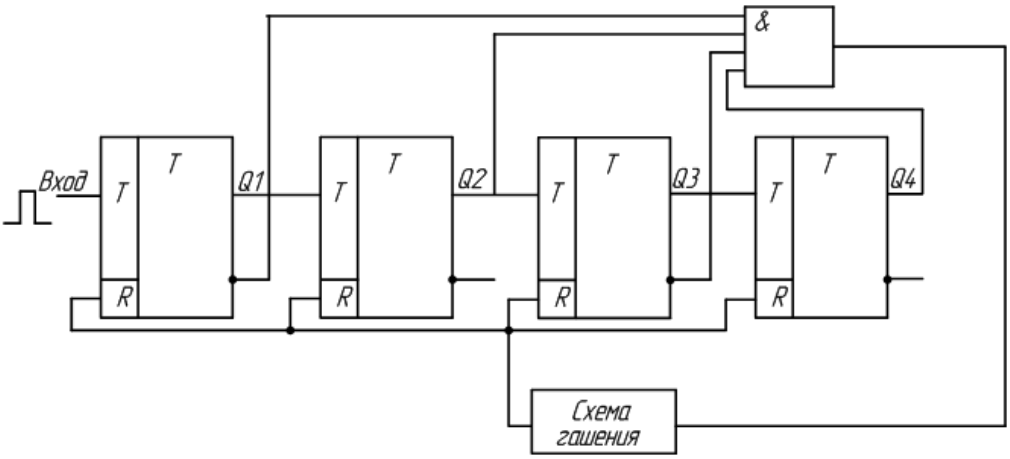


Рисунок 2.5 – Лічильник з модулем лічби М=10

Розглянемо лічильник з М=11 на основі двійкового лічильника в одній мікросхемі (без інверсних виходів).

$11_{10}=1011_2$

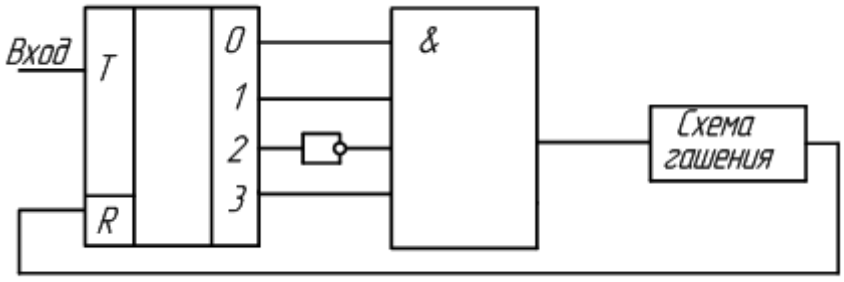


Рисунок 2.6 – Лічильник з модулем лічби М=11

В якості схеми скидання може бути RS-тригер.

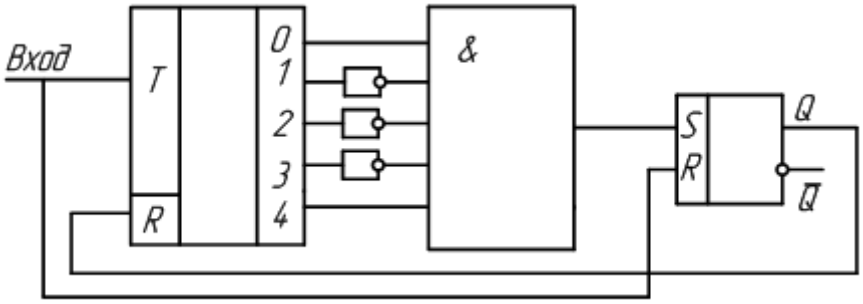


Рисунок 2.7 – Лічильник з модулем лічби М=17

В цій схемі $M=10001_2 = 17_{10}$

Сигнал на вході К лічильника буде діяти протягом одного періоду вхідних імпульсів