НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Кафедра комп'ютерних систем і мереж

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт
з дисципліни
«Архітектура комп'ютера»

для студентів спеціальності 123 «Комп'ютерна інженерія»

всіх форм навчання

2019

Методичні вказівки до лабораторних робіт з дисципліни «Архітектура комп'ютера» для студентів спеціальності 123 «Комп'ютерна інженерія» всіх форм навчання / Укл.: М.Д. Місюра, В.А. Лахно. – Київ: НУБіП, 2019. – 83 с.

Укладачі: М.Д. Місюра, к.т.н., доцент каф. комп'ютерних систем і мереж НУБіП України
В.А. Лахно, д.т.н., професор каф. комп'ютерних систем і мереж НУБіП України

Рецензенти: В.В. Хиленко, д.т.н., професор каф. комп'ютерних наук НУБіП України
В.В. Шкарупило, к.т.н., доцент доцент каф. комп'ютерних систем і мереж НУБіП України

Відповідальний за випуск: М.Д. Місюра, к.т.н., доцент каф. комп'ютерних систем і мереж НУБіП України

Затверджено:
на засіданні кафедри
«Комп'ютерних систем і мережі»
Протокол №_3_ від _31_._10_.2019 р.
Рекомендовано до видання:
НМК факультету ІТ
Протокол №_4_ від _18_._11_.2019 р.

Зміст

ВСТУП

Матеріал, пов’язаний з вивченням CASE-засобів, які використовуються для проектування електронних схем сучасних комп’ютерів, складає значну частину курсу «Архітектура комп’ютера». Дисципліна «Архітектура комп’ютера» є одною з базових в системі знань та вмінь, що формують бакалавра та спеціаліста із спеціальності “Програмна інженерія”. Мета викладення дисципліни «Архітектура комп’ютера» – в сформувати в студентів знання, уміння і навички, необхідні для подальшого використання сучасних апаратних засобів комп'ютерів.

Зважаючи на важливість і обширність матеріалу, пов’язаного з питаннями методів аналізу і синтезу логічних електронних схем комп’ютерів із застосуванням CASE-засобів, його вивчення в сучасній методиці викладання дисципліни «Архітектура комп’ютера», займає провідну позицію.

У даних методичних вказівках, у відповідності до навчального плану, містяться десять лабораторних робіт, які у певній мірі дозволяють студентам оволодіти методами аналізу і синтезу електронних схем сучасних комп’ютерів із допомогою найбільш поширених CASE-засобів.

Ціль методичних вказівок – з метою сприяння вирішенню завдань підготовки спеціалістів у галузі комп’ютерних технологій, допомогти студентам оволодіти практичними навиками роботи з аналізу логічних схем комп’ютерів.

Спробою досягти цієї цілі й визначається структура даних методичних вказівок. Частина лабораторних робот виконується у відповідності до індивідуальних варіантів, а частина робот виконується студентом у відповідності до завдання, яке міститься у розділі – хід роботи.

Після виконання лабораторної роботи кожен студент повинен оформити звіт на аркуші формату А4 якій містить наступні пункти:

- *мета роботи;*
- *варіант завдання;*

- *логічна схема побудована у відповідному середовищі – EWB, MC8, Multisim 9 або Multisim 10;*
- *таблиця із результатами експериментів;*
- *висновок по роботі.*

При захисті роботи студент відповідає на питання, які наведені після кожної лабораторні роботи (питання для самоперевірки).

Критерії оцінювання робот. Оцінюється відсоток виконаних робіт.

A –	"5"	— 4,5 – 5,0	— 91% – 100%
B –	"4"	— 4,0 – 4,4	— 86% – 90%
C –	"4"	— 3,5 – 3,9	— 81% – 85%
D –	"3"	— 3,0 – 3,4	— 76% – 80%
E –	"3"	— 2,5 – 2,9	— 71% – 75%
FX –	"2"	— 2,0 – 2,4	— 66% – 70%
F –	"2"	— 1,5 – 1,9	— 60% – 65%

Методичні вказівки призначені для виконання лабораторних робіт студентами всіх форм навчання 123 «Комп'ютерна інженерія».

Згідно з планованим розподілом навчального часу дисципліну поділено на 4 модулі. Дані методичні вказівки охоплюють перший та другий модулі.

Теми першого модуля: Елементи і вузли ПК (логіка, тригери, регістри, лічильники, шифратори, дешифратори, мультіплексори, демультіплексори, додавачі, електронна пам'ять).

Теми другого модуля. Процесори. Принципи розробки сучасних комп'ютерів. Паралелізм на рівні команд. Структура мікропроцесорної системи (МПС). Структура і робота мікропроцесора. Блоки МПС.

Відповідно, лабораторні роботи з першої по четверту відноситься до першого модуля, лабораторні роботи з п'ятої по десяту до другого модуля.

Лабораторна робота № 1. Вивчення логічних елементів електронних схем у комп'ютерів

Мета роботи: *закріплення теоретичних знань із структури логічних елементів та їх застосування при побудові логічних схем цифрових пристроїв та синтезу сигналів на виході ЦА.*

Логічною схемою називається сукупність логічних електронних елементів, з'єднаних між собою так, щоб виконувався заданий закон функціонування схеми, тобто виконувалася задана логічна функція.

За залежністю вихідного сигналу від вхідного всі електронні логічні схеми можна умовно розподілити на:

- *схеми першого роду*, тобто *комбінаційні схеми*, вихідний сигнал яких залежить лише від стану вхідних сигналів у кожен момент часу;
- *схеми другого роду* або накопичувальні схеми, які містять елементи з пам'яттю, вихідний сигнал яких залежить і від вхідних сигналів, і від стану схеми в попередні моменти часу.

За кількістю входів і виходів схеми бувають:

- з одним входом і одним виходом;
- з кількома входами й одним виходом;
- з одним входом й кількома виходами;
- з кількома входами й виходами.

За способом здійснення синхронізації схеми бувають із *зовнішньою синхронізацією (синхронні автомати)* та *з внутрішньою синхронізацією* (асинхронні автомати є їх окремим випадком).

Практично будь-який комп'ютер складається з комбінації схем першого й другого родів різної складності. Таким чином, основою будь-якого цифрового автомата, що обробляє цифрову інформацію, є електронні елементи двох типів: *логічні* або *комбінаційні* й *запам'ятовуючі*. Логічні елементи виконують найпростіші логічні операції над цифровою інформацією, а запам'ятовуючі слугують для її зберігання.

Технічним аналогом булевої функції в обчислювальній техніці є так звана *комбінаційна схема.* На вхід комбінаційної схеми

надходять, а з виходу знімаються електричні сигнали у вигляді одного з рівнів напруги, що відповідає значенням логічного 0 і логічної 1.

Наприклад, розглянемо схему, яка має *m* входів і *n* виходів (рис. 1.1). На її входи можуть бути подані набори значень вхідних перемінних X_i $\{0,1\}$, $i = \overline{1,m}$, а на виходах формуються вихідні перемінні $Y_j \in \{0,1\}$, $j = \overline{1,n}$.

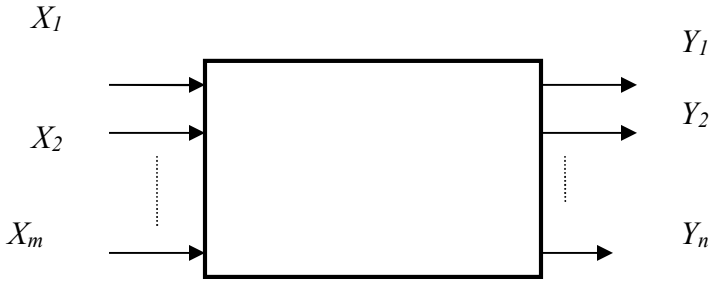


Рис. 1.1. Комбінаційна схема

Схема називається *комбінаційною*, якщо кожну з *n* функцій її виходів $Y_1, Y_2, \ldots, Y_n$ можна представити як булеву функцію вхідних перемінних $X_1, X_2, \ldots, X_m$.

Як випливає з визначення комбінаційної схеми, значення вихідних перемінних Y_j у довільний момент часу однозначно визначаються значеннями вхідних перемінних X_i.

Структурно комбінаційна схема може бути представлена як сукупність елементарних логічних схем – логічних елементів (ЛЕ). ЛЕ виконують над вхідними перемінними елементарні логічні операції типу **„І-НЕ", „І", „АБО", „АБО-НЕ"** і т. ін.

Кількість входів логічного елемента відповідає кількості аргументів відтворюваної ним булевої функції. Графічне зображення комбінаційної схеми, при якому показано зв'язки між різними елементами, а самі елементи представлено умовними позначками, називається *функціональною схемою.*

Під час розробки комбінаційних схем доводиться розв'язувати задачі аналізу й синтезу.

Задача аналізу полягає у визначенні статичних і динамічних властивостей комбінаційної схеми. У статиці визначаються булеві

функції, реалізовані комбінаційною схемою за відомою їй структурою. У динаміці розглядається здатність надійного функціонування схеми в перехідних процесах при зміні значень перемінних на входах схеми, тобто визначається наявність на виходах схеми можливих небажаних імпульсних сигналів, які не випливають безпосередньо з виразів для булевих функцій, реалізованих схемою.

Задача синтезу полягає в побудові із заданого набору логічних елементів комбінаційної схеми, яка реалізує задану систему булевих функцій.

Розв'язання задачі синтезу не є однозначним, можна припустити різні варіанти комбінаційних схем, які реалізують одну й ту саму систему булевих функцій, але відрізняються за тими або іншими параметрами. Розробник комбінаційних схем з цієї множини варіантів вибирає один, виходячи з додаткових критеріїв: мінімальної кількості логічних елементів, необхідних для реалізації схеми, максимальної швидкодії й т. ін. Існують різні методи синтезу комбінаційних схем, серед яких найбільше розроблено канонічний метод.

Елементи, які реалізують найпростіші логічні функції, схематично представляють у вигляді прямокутників, на полі яких зображують символ, що позначає функцію, виконувану цим елементом. Вхідні перемінні прийнято зображувати зліва, а вихідні – справа. Вважається, що передача інформації відбувається зліва направо.

Якщо виходи одних елементів з'єднати з входами інших, то отримаємо схему, яка реалізує більш складну функцію. Сукупність різних типів елементів, достатніх для відтворення будь-якої логічної функції, назвемо логічним базисом. Елементи **„І"** й **„НЕ"** представляють такий логічний базис. Елемент типу **„АБО"** може бути отримано з'єднанням елементів **„І"** та **„НЕ".**

Логічний базис може складатися лише з одного типу елементів, наприклад елемента типу **„І-НЕ"**, схему якого показано в табл. 1.1. Універсальність елемента **„І-НЕ"** забезпечила йому широке застосування у створенні обчислювальної техніки. Існують й інші елементи, які реалізують найпростіші логічні функції. До них, наприклад, належать елементи сумування за модулем два, які реалізують функцію нерівнозначності двох перемінних.

Зазначимо, що кожен реальний логічний елемент має деякий час затримки зміни вихідного сигналу по відношенню до вхідного. Однак у комбінаційних схемах при їх формальному описі часом затримки логічних елементів нехтують.

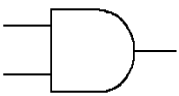
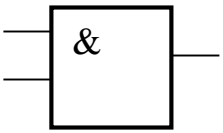
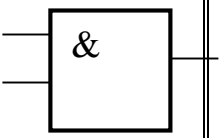
Як уже було відзначено, прості (елементарні) логічні функції апаратно реалізуються за допомогою відповідних комбінаційних електронних логічних елементів – *вентилів*. Можна вважати, що ці елементарні логічні функції є *логічними операторами* згаданих електронних елементів, тобто схем. Кожну таку схему позначають певним графічним символом, який може бути орієнтованим або ж неорієнтованим. Наведемо приклади графічного позначення цих схем (див. табл. 1.1).

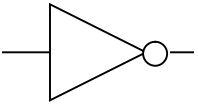
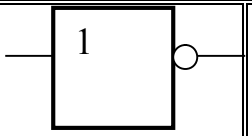
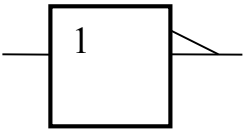
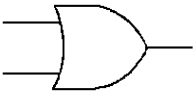
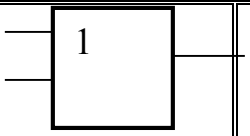
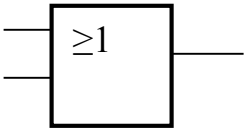
Аналізуючи або синтезуючи логічні ланцюги, слід ураховувати таку обставину.

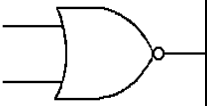
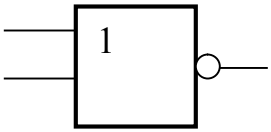
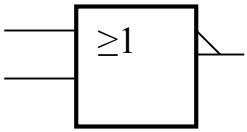
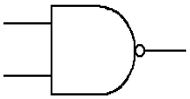
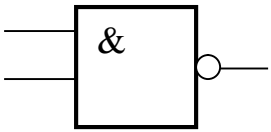
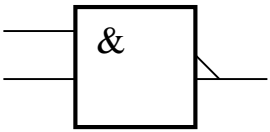
Здійснити логічні функції на практиці дозволяють різноманітні так звані логічні (цифрові) напівпровідникові схеми – вентилі, вихідні сигнали яких однозначно визначено комбінаціями рівнів сигналів на входах цих схем. Причому, і вхідні, і вихідні сигнали цих вентилів можуть бути імпульсними або потенційними й мають два фіксовані значення: високий (H) або низький (L) рівень. Коли логічній „1” відповідає високий рівень, тоді логічному „0” – низький.

Таблиця 1.1

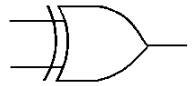
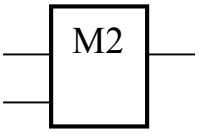
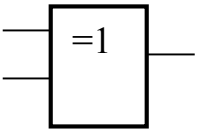
Приклади графічного зображення елементів логічних схем

Традиційне	**Традиційне**	**IEEE/ANSI**
Схема, яка реалізує логічну функцію **„І”**		
 $y = x_1 \wedge x_2$		
Схема, яка реалізує елементарну логічну функцію **„НЕ”**		

 $y = \overline{x}_1$		
Схема, яка реалізує елементарну логічну функцію **„АБО”**		
 $y = x_1 \vee x_2$		

Схема, яка реалізує елементарну логічну функцію **„АБО-НЕ”** (стрілка Пірса)		
 $y = \overline{x_1 \vee x_2}$		
Схема, яка реалізує логічну функцію **„І-НЕ”**(штрих Шеффера)		
 $y = \overline{x_1 \wedge x_2}$		

Продовження - Таблиця 1.1

Схема, яка реалізує логічну функцію **„АБО, що виключає”** (тобто суму за модулем 2)		
 $y = x_1 \otimes x_2 =$ $\bar{x}_2 x_1 + \bar{x}_1 x_2$		

Будь-яка комбінаційна схема являє собою з’єднання логічних елементів, в якому входи деяких елементів підключаються до входів інших. Але ж не кожне з’єднання логічних елементів відноситься до класу комбінаційних схем. З’єднання логічних елементів, в якому сигнал, який поступає на вхід деякого елементу, проходячи через один чи декілька елементів, надходить на вхід того ж самого елементу, не належить до класу комбінаційних схем.

На рис. 1.2 – 1.5 наведено схеми, таблиці істинності, діаграми роботи логічних елементів.

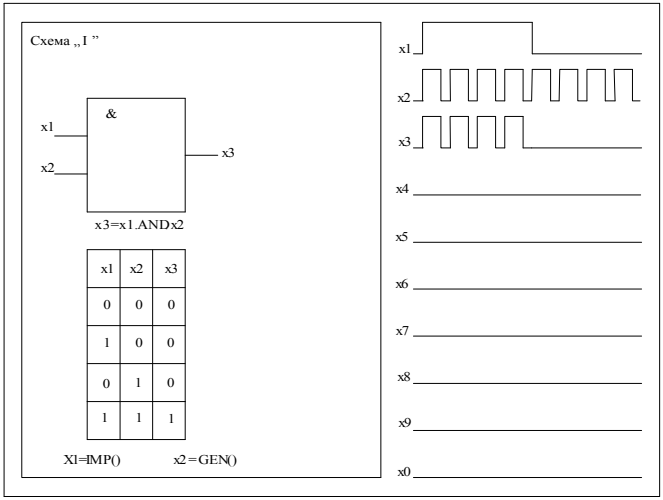


Рис. 1.2. Схема, таблиця істинності, діаграма роботи логічного елементу “І”

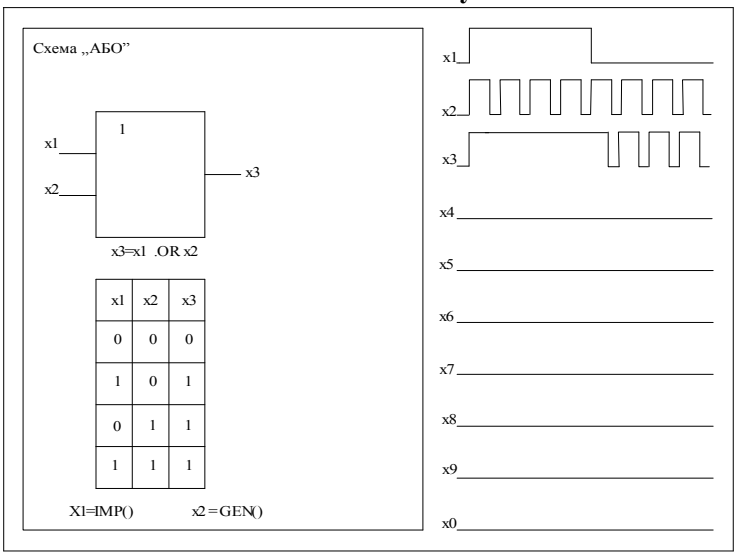


Рис. 1.3. Схема, таблиця істинності, діаграма роботи логічного елементу “АБО”

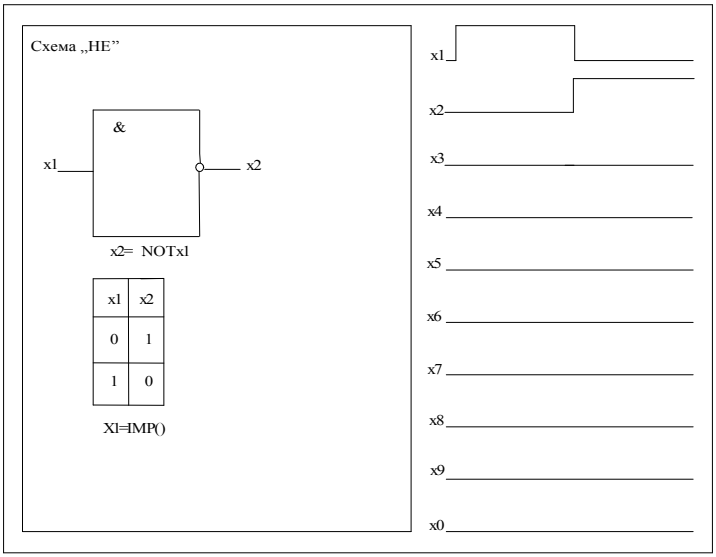


Рис. 1.4. Схема, таблиця істинності, діаграма роботи логічного елементу “НЕ”

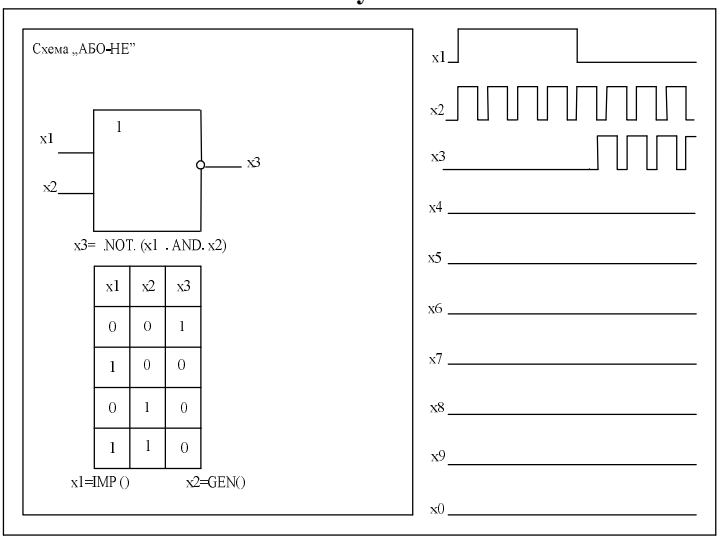
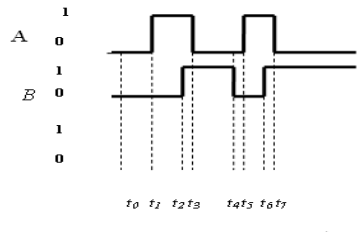
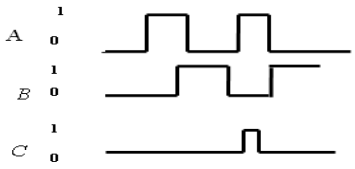


Рис. 1.5. Схема, таблиця істинності, діаграма роботи логічного елементу “АБО-НЕ”

Завдання. Побудувати вихідний сигнал для логічних елементів «І», «НЕ», «АБО», «І-НЕ», «АБО-НЕ» рис. 1.6 а) та б).



а)



б)

Рис. 1.6. Тимчасові діаграми

Розглянемо приклад, в якому спробуємо спроектувати логічну схему п'яти бітового суматора, що використовує окремі повні суматори. Сума з'являється на виходах S_4, S_3, S_2, S_1, S_0.

Нижче показана таблиця істинності пристрою, що має три входи *A, B,* C_{in}, а також два виходи: *S* і C_{out}. Всього для трьох входів можливі вісім наборів станів, при цьому для кожного конкретного випадку в таблиці істинності 1.2 і на рис. 1.7, 1.8 вказані можливі вихідні сигнали.

Таблиця 1.2

Таблиця істинності повного суматора

Вхід бітів першого доданку	**Вхід Бітів другого доданку**	**Вхід бітів перенесення**	**Вихід бітів суми**	**Вихід бітів перенесення**
A	**B**	**Свх**	**S**	**Вих.**
0	0	0	0	0
0	0	1	1	0

Вхід бітів першого доданку	**Вхід Бітів другого доданку**	**Вхід бітів перенесення**	**Вихід бітів суми**	**Вихід бітів перенесення**
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

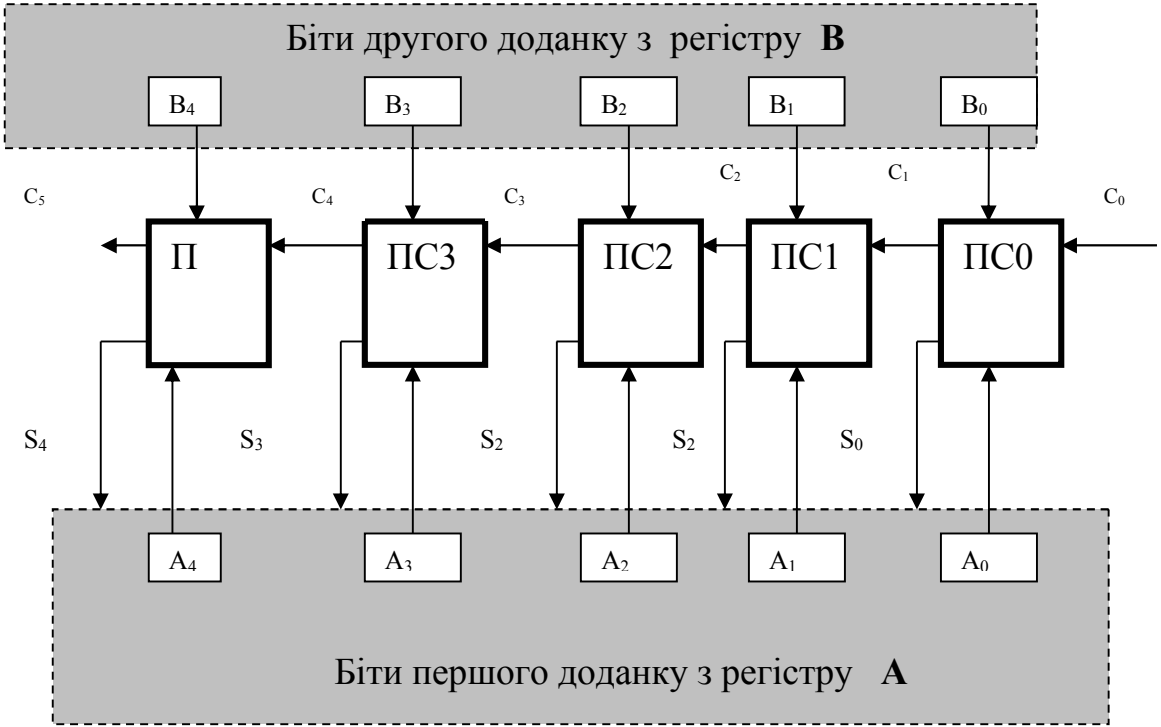


Рис. 1.7. Блок схема паралельного п’яти бітового суматора, що використовує окремі повні суматори

Пристрій називається паралельним суматором, тому, що всі біти чисел, що складаються, подаються на входи схеми **одночасно.** Це значить, що в кожному розряді складання виконується за один такт. Це відрізняється від звичного складання на папері, коли ми складаємо біти (цифри) по черзі, т.ч. паралельне складання виконується швидше.

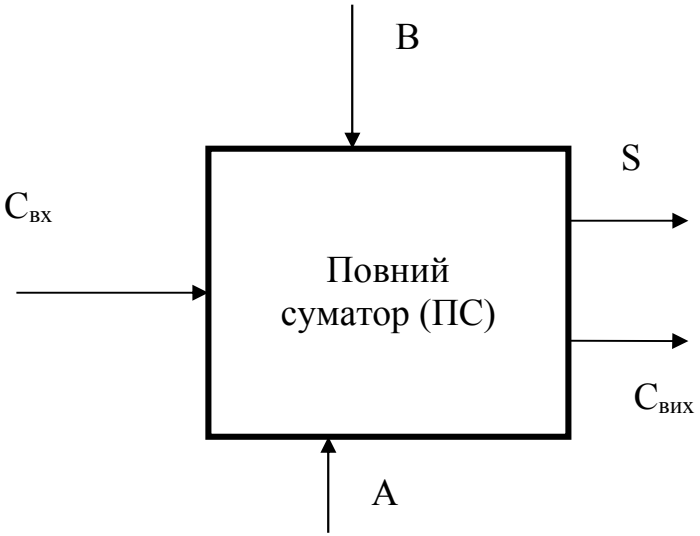


Рис. 1.8. Схема суматора

Розглянемо, наприклад, випадок коли $A=0,\ B=0,\ C_{in}=1$. Повний суматор повинен скласти ці біти і одержати суму S, рівну 0, і перенесення C_{out}, рівне 1.

Схема має два виходи, тому можна проектувати логіку для кожного виходу по окремості. Почнемо з виходу S. Таблиця істинності показує, що можливі чотири ситуації, коли $S=1$. Можливо записати вираз для S, використовуючи диз'юнктивну форму:

$$S = \overline{A} \cdot \overline{B} \cdot C_{in} + \overline{A} \cdot B \cdot \overline{C}_{in} + A \cdot \overline{B} \cdot \overline{C}_{in} + A \cdot B \cdot C_{in}.$$

Спробуємо спростити цей вираз, тобто винесемо за дужки загальні множники. Оскільки жодна пара доданків не має двох

загальних змінних, винесемо за дужки з перших двох членів множник $\overline{A}$, а з двох останніх A. В результаті одержимо:

$$S = \overline{A} \cdot (\overline{B} \cdot C_{in} + B \cdot \overline{C}_{in}) + A \cdot (\overline{B} \cdot \overline{C}_{in} + B \cdot C_{in}).$$

Перший член, записаний в дужках, є виразом виключає «**АБО»** для змінних B *и* C_{in}. Його можна записати у вигляді $B \otimes C_{in}$. Другий член в дужках – це операція **«АБО-НІ», що виключає**, для тих же змінних B *и* C_{in}. Його можна записати у вигляді $\overline{B \otimes C_{in}}$. Тоді вираз для S отримає наступний вигляд: $S = \overline{A} \cdot (B \otimes C_{in}) + A \cdot (\overline{B \otimes C_{in}})$.

Замінимо $B \otimes C_{in}$ на *X. Отже* $S = \overline{A} \cdot X + A \cdot \overline{X} = A \otimes X$, це операція **«АБО» що виключає.** Проведемо зміну $S = A \otimes [B \otimes C_{in}]$. Отже розглянемо сигнали на виході C_{out}*, які наведені у* таблиці істинності. Запишемо вираз для сигналу C_{out} у диз'юнктивній формі:

$$C_{out} = \overline{A} \cdot B \cdot C_{in} + A \cdot B \cdot \overline{C}_{in} + A \cdot B \cdot \overline{C}_{in} + A \cdot B \cdot C_{in}.$$

Спростимо вираз

$$C_{out} = B \cdot C_{in} \cdot (\overline{A} + A) + A \cdot C_{in} \cdot (\overline{B} + B) + A \cdot B \cdot (\overline{C}_{in} + C_{in}) = B \cdot C_{in} + A \cdot C_{in} + A \cdot B.$$

Подальша мінімізація одержаного виразу неможлива. Схема одержана в результаті виглядатиме таким чином, див. рис. 1.9.

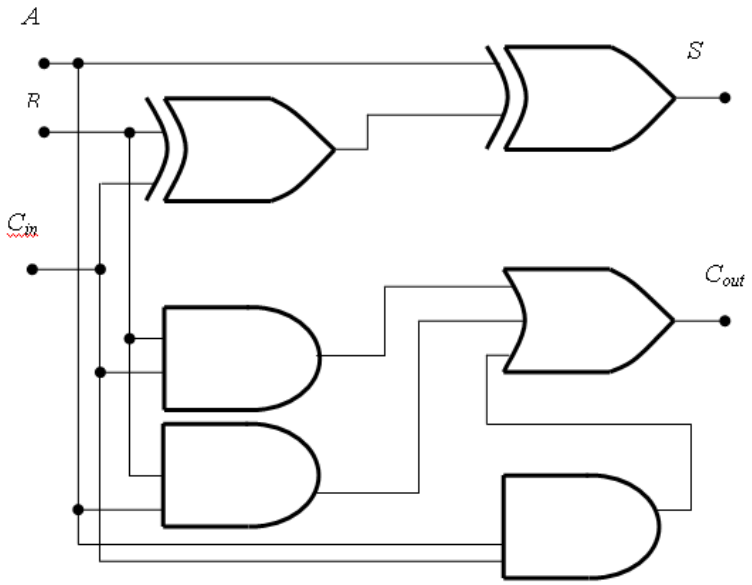


Рис. 1.9. Схема п’яти бітового суматора

Хід виконання роботи

1. Вивчити теоретичні відомості по роботі. У ході виконання роботи необхідно вести протокол роботи, куди рекомендується заносити схеми логічних елементів і діаграми їх функціонування, логічні функції, таблиці істинності.

2. Побудувати логічну схему у відповідності до наступної таблиці істинності.

Таблиця 1.3

Таблиця істинності для побудови логічного виразу

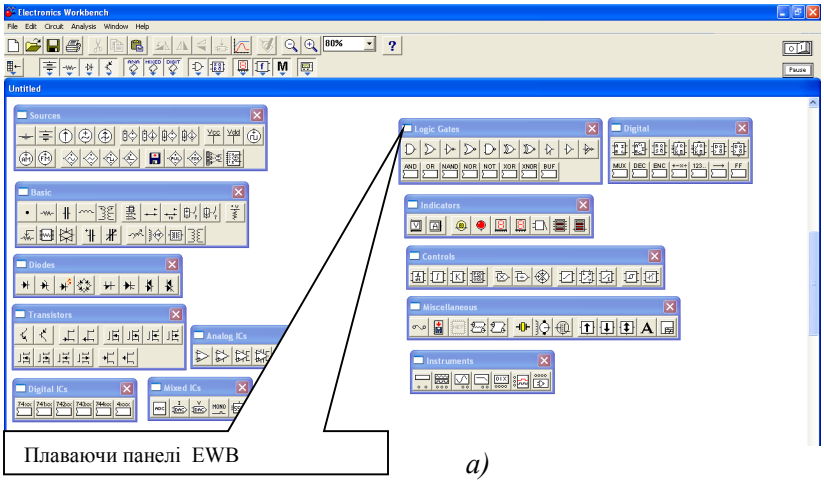
X1	X2	X3	У
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Лабораторна робота № 2. Експериментальні дослідження комбінаційних схем у програмах Electronics Workbench та Multisim

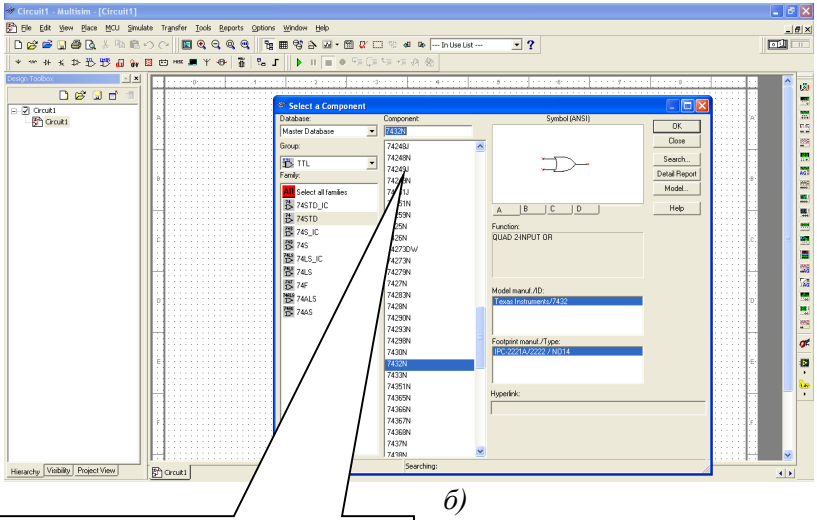
Мета роботи: Навчитися моделювати роботу комбінаційних схем пам'яті у програмах Electronics Workbench та Multisim.

Системи схемотехнічного моделювання *Electronics Workbench* та *Multisim* призначена для моделювання і аналізу електричних схем, див. рис. 2. 1.

Системи *Electronics Workbench* (рис. 2.1 а) та *Mulsim* (рис. 2.1 б) дозволяють моделювати аналогові, цифрові і цифро-аналогові схеми великого ступеня складності. Бібліотеки, що є в програмах, включають великий набір поширених електронних компонентів. Є можливість підключення і створення нових бібліотек компонентів.



а)



б)

Рис. 2.1. Загальний вигляд Electronics Workbench та Mulsim 10

Параметри компонентів можна змінювати в широкому діапазоні значень. Прості компоненти описуються набором параметрів, значення яких можна змінювати безпосередньо з

клавіатури, активні елементи – у вигляді моделі що описує конкретний елемент або його ідеальне уявлення. Модель вибирається із списку бібліотек компонентів, параметри моделі також можуть бути змінені користувачем. Широкий набір приладів дозволяє проводити вимірювання різних величин, задавати вхідні дії, будувати графіки. Всі прилади зображаються у вигляді, максимально наближеному до реального. Результати моделювання можна вивести на принтер або імпортувати в текстового або графічного редактора для їх подальшої обробки. Програми *Electronics Workbench (Mulsim)* сумісні з програмою *P-SPICE,* тобто надає можливість експорту і імпорту схем і результатів вимірювань в різні її версії. Працюючи з *Electronics Workbench (Mulsim)*, експериментатор застрахований від випадкової поразки струмом, а прилади не вийдуть з ладу через неправильно зібрану схему. Завдяки цій програмі у розпорядженні користувача є такий широкий набір приладів, який навряд чи буде доступний в реальному житті. Таким чином, у Вас завжди є унікальна можливість для планування і проведення широкого спектру досліджень електронних схем при мінімальних витратах часу.

Всі операції проводяться за допомогою миші і клавіатури. Управління тільки з клавіатури неможливе. Шляхом настройки приладів можна: змінювати шкали приладів залежно від діапазону вимірювань, задавати режим роботи приладу, задавати вид вхідних дій на схему (постійні і гармонійні струми і напруги, трикутні і прямокутні імпульси). Графічні можливості програми дозволяють: одночасно спостерігати декілька кривих на графіку, відображати криві на графіках різними кольорами, вимірювати координати крапок на графіку, імпортувати дані в графічного редактора, що дозволяє провести необхідні перетворення малюнка і висновок його на принтер. *Electronics Workbench* дозволяє використовувати результати, одержані в програмах *P-SPICE, PCB*, а також передавати результати з *Electronics Workbench* в ці програми. Можна вставити схему або її фрагмент в текстового редактора і надрукувати в ньому пояснення або зауваження по роботі схеми.

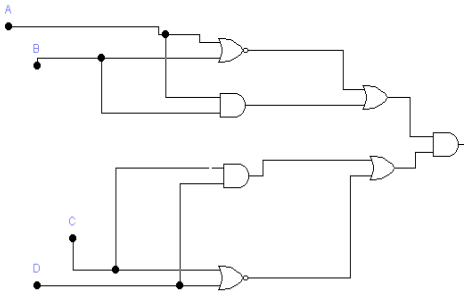
При побудові і редагуванні схем виконуються наступні операції:

• вибір компоненту з бібліотеки компонентів;

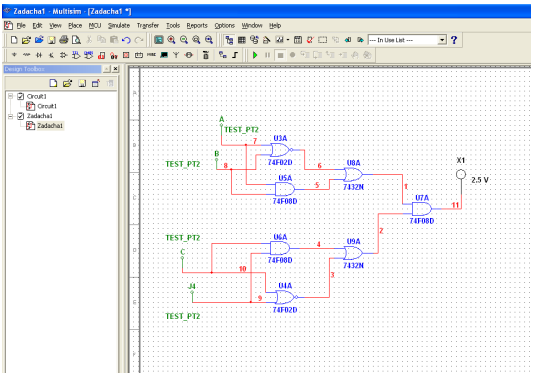
• виділення об'єкту;
• переміщення об'єкту;
• копіювання об'єктів;
• видалення об'єктів;
• з'єднання компонентів схеми провідниками;
• установка значень компонентів;
• підключення приладів.

Якщо схема не поміщається на екрані монітора, будь-яку її ділянку можна проглянути за допомогою лінійок прокрутки, розташованих справа і під робочим полем. Після побудови схеми і підключення приладів аналіз її роботи починається після натиснення вимикача в правому верхньому кутку вікна програми.

Експеримент 1. Знайдіть аналітичний вираз функції, яка реалізується схемою, приведеною на рис. 2.2. Зберіть схему у *Electronics Workbench* (рис. 2.2. а) або *Mulsim* (рис. 2.2. б), підключіть входи *D, C, B, A* до джерела логічних сигналів, а вихід — до логічного пробника. Включіть схему і перевірте правильність аналітичного виразу.



а)



б)

Рис. 2.2. Схема до експерименту №1

Експеримент 2. Викличте генератор слів і логічний аналізатор. Запрограмуйте генератор на формування послідовності чотирьох розрядних слів, відповідних числам натурального ряду від 0 до 15. Підключіть його виходи до відповідних входів схеми, приведеної на рис. 2.2. (*A* - молодший розряд числа, *D* - старший). Досліджуйте роботу схеми в режимах *"STEP"* та *"CYCLE"*.

Намалюйте тимчасові діаграми сигналів на виходах всіх логічних елементів схеми для всіх можливих комбінацій вхідних сигналів. Перевірте правильність виконання завдання за допомогою логічного аналізатора.

Експеримент 3. Проведіть аналіз роботи схеми, зображеної на рис. 2.3, для чого складіть таблиці реалізованих функцій, якщо сигнал в крапці 1 сприймається елементом **«АБО»**:

а) як логічна 1;

б) як логічний 0.

Виберіть необхідні інструменти для проведення експериментальної перевірки схеми і визначте, як сприймається сигнал на непідключеному вході при роботі базових елементів.

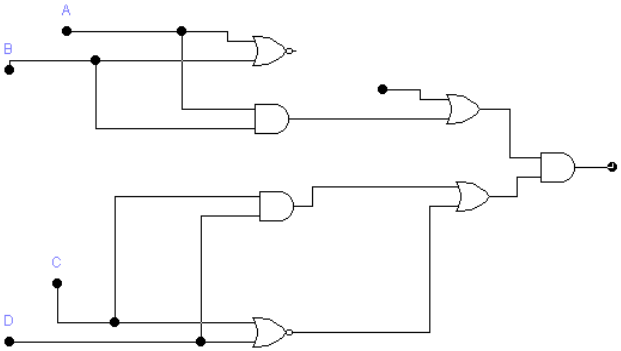


Рис. 2.3. Схема до експерименту № 3

Експеримент 4. Проведіть аналіз роботи логічного пристрою, зібраного на мікросхемах *7404* і *7410*, показаного на рис. 2.4. Визначте, яку математичну операцію виконує даний пристрій, якщо комбінації логічних рівнів на вході розглядати як числа. Зберіть схему, підключіть необхідні прилади і проведіть експериментальне дослідження роботи схеми.

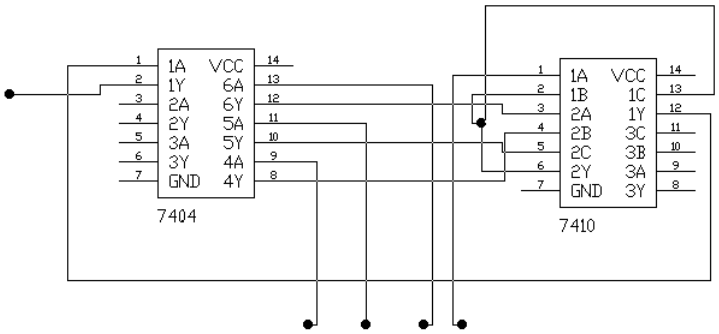
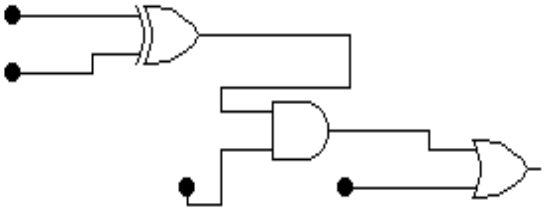


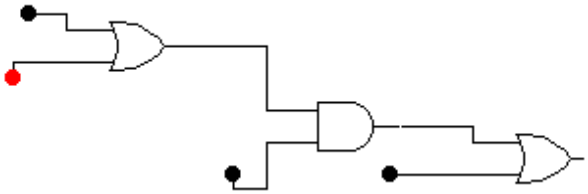
Рис. 2.4. Схема до експерименту № 4

Експеримент 4. При монтажі схеми, приведеної на рис. 2.5 а, була допущена помилка: замість елемента **XOR** був

використаний елемент **OR** (рис. 2.5 б). Знайдіть комбінації вхідних сигналів, які дозволяють виявити помилку монтажника.



а)



б)

Рис. 2.5. Схема до експерименту № 5

Експеримент 5. Спростить схему приведену на рис. 2.6. Перевірте еквівалентність сигналів на виході попередньої схеми та отриманої при спрощені.

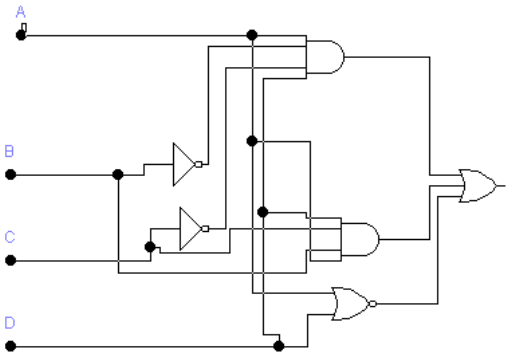


Рис. 2.6. Схема до експерименту № 6

Хід виконання роботи

1. Повторити конспект (теоретичні відомості).
2. Отримати допуск до роботи.
3. Скласти у *Electronics Workbench* або *Mulsim* відповідні схеми.
4. Провести відповідні експерименти. Заповнити протокол та скласти звіт по лабораторній роботі.
5. Зробити висновки по роботі.
6. Відповісти на запитання для самоперевірки.

Запитання для самоперевірки

1. Чим відрізняється робота у *Electronics Workbench* та *Mulsim*?
2. За допомогою яких вікон або панелей побудувати комбінаційну схему у *Electronics Workbench* та *Mulsim*?
3. Як перевірити працездатність схеми у *Electronics Workbench* або *Mulsim*?
4. За допомогою яких інструментів можливо спростити схему у *Electronics Workbench* або *Mulsim*?

Лабораторна робота № 3. Експериментальні дослідження тригерів у програмах Electronics Workbench та Multisim

Мета роботи: Навчитися моделювати роботу цифрових пристроїв з елементами пам'яті у програмах Electronics Workbench та Multisim.

Тригер має два стійкі стани: *Q=1* і *Q=0*, тому його іноді називають бістабільною схемою. В якому з цих станів опиниться тригер, залежить від сигналів на входах тригера і від його

попереднього стану, тобто він має пам'ять. Можна сказати, що тригер є елементарним елементом пам'яті. Тип тригера визначається алгоритмом його роботи. Залежно від алгоритму роботи, тригер може мати настановні, інформаційні і управляючі входи. Настановні входи встановлюють стан тригера незалежно від стану інших входів. Входи управління дозволяють запис даних, що подаються на інформаційні входи. Найпоширенішими є тригери *RS, JK, D* і *T*- типів.

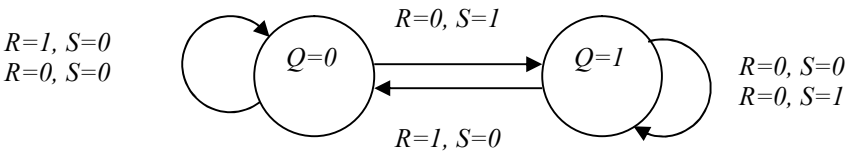
1. Тригер типа RS

RS-тригер - найпростіший автомат з пам'яттю, який може знаходитися в двох станах. Тригер має два входи: установки S (SET - установка) і скидання R (RESET - скидання), на які подаються вхідні сигнали від зовнішніх джерел. При подачі на вхід установки активного логічного рівня тригер встановлюється в 1 ($Q = 1, \quad \overline{Q} = 0$), при подачі активного рівня на вхід скидання тригер встановлюється в 0 ($Q = 0, \quad \overline{Q} = 1$). Якщо подати на обидва входи установки (збудження) пасивний рівень, то тригер зберігатиме попередній стан виходів: $Q = 0 \quad (\overline{Q} = 1) \quad$ *або* $\quad Q = 1 \quad (\overline{Q} = 0)$

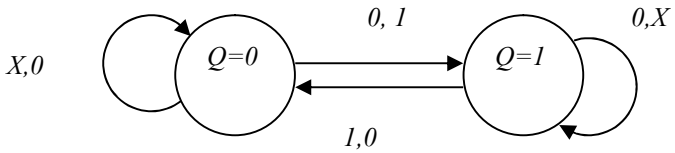
Опис роботи RS-тригера можна доповнити графом рис. 3.1 (графічний спосіб).

Графік на рис. 3.1 а показує, що схема, яка знаходилася в стані *Q=0*, зберігає цей стан як при дії вхідного набору *R=0*, *S=0*, так і при дії *R=1*, *S=0*. Якщо ж на вхід схеми, що знаходиться в стані *Q=0*, подіяти набором *R=0*, *S=1*, то вона переходить в стан *Q=1* і зберігає його при вхідних наборах *R=0*, *S=1*, або *R=0*, *S=0*.

На рис. 3.1 б той же граф тригера намальований більш компактно. Вхідні сигнали, які можуть приймати будь-які значення (як *0*, так і *1*), позначені як *X*, а позиція позначення відповідає послідовності *R*, *S*.



а)



б)

Рис. 3.1. Граф роботи RS-тригера

2. JK- тригер

Тригер *JK*- типу має складнішу, в порівнянні з *RS*-тригером, структуру і більш широкі функціональні можливості. Крім інформаційних входів *J* і *K* і прямого та інверсного виходів, *JK*-тригер має вхід управління *C* (цей вхід також називають тактуючим або рахунковим), а також асинхронні *R* і *S* - входи. Входи *R* і *S* мають пріоритет над іншими. Активний рівень сигналу на вході *S* встановлює тригер в стан 0=1, а активний рівень сигналу на вході R - в стан *Q=0*, незалежно від сигналів на решті входів.

3. D-тригер.

D-тригер має один інформаційний вхід D (data - дані). Інформація з входу *D* заноситься в тригер по позитивному

перепаду імпульсу на рахунковому вході *C* і зберігається до наступного позитивного перепаду на рахунковому вході тригера. Крім рахункового *C* і інформаційного *D* входів, тригер забезпечений асинхронними *R* і *S* входами. На рис. 3.2 зображено граф роботи D тригера.

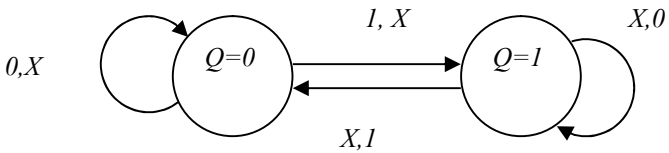


Рис. 3.2. Граф роботи D-тригера

Експеримент 1. Зберіть схему зображену на рис. 3.3. Включіть схему. Послідовно подайте на схему наступні сигнали: *S=1, R=0; S=0, R=0*. Поясніть отримані результати.

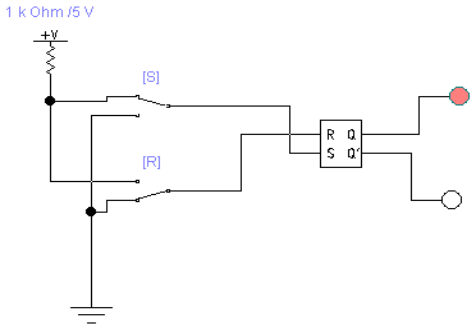


Рис. 3.3. Дослідження RS-тригера

Експеримент 2. Дослідження *JK*-тригера, побудованого на базі логічних елементів **«І»** *RS*-тригерів.

Зберіть схему зображену на рис. 3.4. Включіть схему. Змінюючи рівень сигналу на вході *C*, складіть тимчасові діаграми сигналів на виходах *Q1* і *Q2* обох *RS*-тригерів і замалюйте їх в розділі "Результати експериментів". Вкажіть режим роботи тригера. Визначте моменти зміни сигналів *Q1* і *Q2* по відношенню до моментів зміни сигналу *C*. Відобразить відмінність в перемикання RS-тригерів на діаграмах.

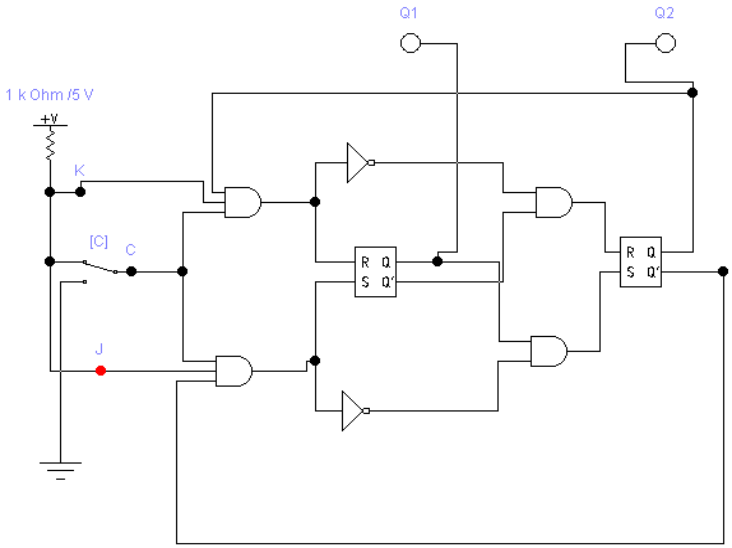


Рис. 3.4. Дослідження *JK*-тригера, побудованого на базі логічних елементів «І» *RS*-тригерів

Експеримент 3. Дослідження роботи D-тригера в рахунковому режимі.

Зберіть схему, зображену на рис. 3.5. Подаючи на рахунковий вхід *C* тактові імпульси за допомогою ключа *[C]* і визначаючи стан виходів тригера за допомогою пробників, складіть тимчасові діаграми роботи тригера в рахунковому режимі і занесіть їх в розділ "Результати експериментів".

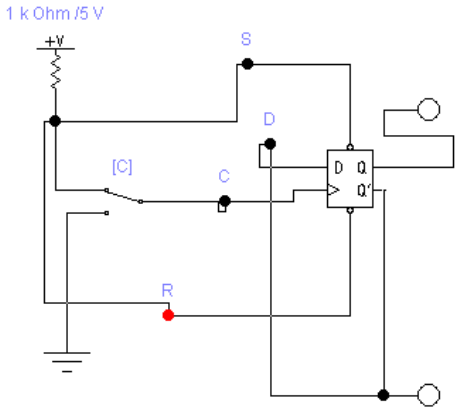


Рис. 3.5. Дослідження роботи D-тригера в рахунковому режимі

Експеримент 4. Дослідження *JK*-тригера в рахунковому режимі (Т-тригер). Зберіть схему, зображену на рис. 3.6. Включіть схему. Змінюючи стан входу *C* відповідним ключем, замалюйте в розділі "Результати експериментів" діаграми роботи тригера в рахунковому режимі.

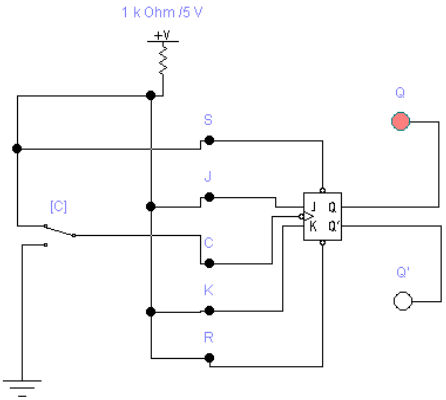


Рис. 3.6. Дослідження роботи JK - тригера в рахунковому режимі

Хід виконання роботи

1. Повторити конспект (теоретичні відомості).
2. Отримати допуск до роботи.
3. Скласти у *Electronics Workbench* або *Mulsim* відповідні схеми.
4. Провести відповідні експерименти. Заповнити протокол та скласти звіт по лабораторній роботі.
5. Підключити до схем осцилограф та замалювати в звіті відповідні тимчасові діаграми роботи тригерів.
6. Зробити висновки по отриманим діаграмам.
7. Відповісти на запитання для самоперевірки.

Запитання для самоперевірки

1. Чим відрізняється робота *RS*-тригера з прямими входами від роботи *RS*-тригера з інверсними входами?
2. Чому комбінація сигналів *11* на входах *RS*-тригера називається «забороненою»?
3. В чому відмінність таблиці переходів тригера від таблиці функцій збудження?
4. Як властивість запам'ятовування відображається в характеристичних рівняннях тригерів?
5. В чому принципова відмінність роботи синхронних тригерів від асинхронних?
6. Яка пріоритетність інформаційних і настановних входів в синхронних тригерах?
7. Чому *JK*-тригер при *J=K=1* не перетворюється на автогенератор?
8. Чому *T*-тригер одержав назву рахункового? Яке число імпульсів він може злічити?
9. Як працює *D*-тригер, якщо *D=Q*?
10. Намалюйте граф-схему роботи *JK*-тригера.
11. Намалюйте тимчасові діаграми роботи відповідно *D, RS* та *JK* тригерів.

12. Намалюйте умовні зображення відповідно *D, RS* та *JK* тригерів.

Лабораторна робота № 4. Експериментальні дослідження лічильників та регістрів у програмах Electronics Workbench та Multisim

Мета роботи: Навчитися моделювати роботу лічильників у програмах Electronics Workbench та Multisim.

Лічильник - пристрій для підрахунку числа вхідних імпульсів. Число, що представляється станом його виходів по фронту кожного вхідного імпульсу, змінюється на одиницю. Лічильник можна реалізувати на декількох тригерах. В лічильниках, що підсумовують, кожний вхідний імпульс збільшує число на його виході на одиницю, у піднімаючих лічильниках кожний вхідний імпульс зменшує це число на одиницю. Найпростіші лічильники - двійкові. При цьому тригери з'єднуються послідовно. Вихід кожного тригера безпосередньо діє на тактовий вхід наступного. Число, утворюване станом інверсних виходів тригерів лічильника, пов'язано з числом освіченим станом прямих виходів тригерів наступним співвідношенням: $N_{пр} = 2^n - N_{інв} - 1,$ де n - розрядність виходу лічильника.

Лічильники характеризуються числом станів протягом одного періоду (циклу). Число станів називають коефіцієнтом перерахунку $K_{під}$, який дорівнює відношенню числа імпульсів N_c на вході до числа імпульсів $N_{Qст}$ на виході старшого розряду за період:

$$K_{під} = \frac{N_c}{N_{Q_{ст}}}.$$

Для побудови підсумовуючого лічильника із $K_{під}=5$ треба, щоб після формування останнього числа з послідовності *{0, 1, 2, 3, 4}* лічильник переходив не до числа 5, а до числа 0. В двійковому коді це означає, що від числа 100 потрібно перейти до числа 000, а не 101. Зміна природного порядку рахунку можлива при введенні додаткових зв'язків між тригерами лічильника. Можна скористатися наступним способом: як тільки лічильник потрапляє в неробочий стан (в даному випадку 101), цей факт повинен бути впізнано і спричинити подальше вироблення сигналу, який перевів би лічильник в стан 000. Розглянемо цей спосіб більш детально. Факт попадання лічильника в неробочий стан описується логічним рівнянням:

$$F = (101) \vee (110) \vee (111) = Q_3 \cdot \overline{Q}_2 \cdot Q_1 \vee Q_3 \cdot Q_2 \cdot \overline{Q}_1 \vee Q_3 \cdot Q_2 \cdot Q_1 =$$
$$= Q_3 \cdot Q_1 \vee Q_3 \cdot Q_2.$$

Стани 110 і 111 також є неробочими і тому враховані при складанні рівняння. Якщо на виході еквівалентної логічної схеми *F=0*, це означає, що лічильник знаходиться в одному з робочих станів: $0 \vee 1 \vee 2 \vee 3 \vee 4$. Як тільки він потрапляє в один з неробочих станів $5 \vee 6 \vee 7$, формується сигнал *F=1*. Поява сигналу *F=1* повинне переводити лічильник в початковий стан 000, отже, цей сигнал потрібно використовувати для дії на входи тригерів лічильника, які здійснювали б скидання лічильника в стан *Q1=Q2=Q3=0*. При реалізації лічильника на тригерах з входами установки логічним нулем для скидання тригерів вимагається подати на входи скидання сигнал *R=0*. Для виявлення факту попадання в неробочий стан використовуємо схему, що реалізовує функцію *F* і виконану на елементах **«І-НЕ»**. Для цього спростимо вираз для функції *F*:

$$F = \overline{Q_3 \cdot Q_1 \vee Q_3 \cdot Q_2} = \overline{Q_3 \cdot (\overline{Q_1} \vee \overline{Q_2})}.$$

Відповідна схемна реалізація приведена на рис. 4.1.

Лічильник працюватиме наступним чином: при рахунку від 0 до 4 все відбувається як в звичайному підсумовуючому лічильнику, із $K_{під}=8$. Встановлюючи сигнали дорівнюють одиниці

та порядку рахунку {0, 1, 2, 3, 4} не перешкоджають. Рахунок відбувається по позитивному фронту імпульсу на рахунковому вході *C*. В той момент, коли лічильник знаходиться в стані 4 (100), наступний тактовий імпульс спочатку переводить лічильник в стан 5 (101), що негайно (задовго до приходу наступного тактового імпульсу) приводить до формування сигналу скидання, яке поступає на встановлюючи входи скидання тригерів. В результаті лічильник скидається в 0 і чекає приходу наступного тактового імпульсу на рахунковий вхід. Один цикл рахунку закінчився, лічильник готовий до початку наступного циклу. Застосовуючи такі схеми із зворотним зв'язком для скидання лічильника, потрібно мати на увазі, що операція скидання займає кінцевий час, тому безпосередньо перед скиданням лічильника в 0 на виході першого тригера з'являються короткочасні імпульси. Це не має значення при підключенні лічильника напряму до індикатора, але при використовуванні цього виходу лічильника як джерела тактових імпульсів можуть виникнути певні проблеми. Схема, в якій це явище усунено, приведена на рис. 4.2. Важливою відмінністю є те, що схема знаходить не факт попадання в неробочий стан 101, а факт попадання в стан 100 і в наступному такті виробляє сигнал скидання.

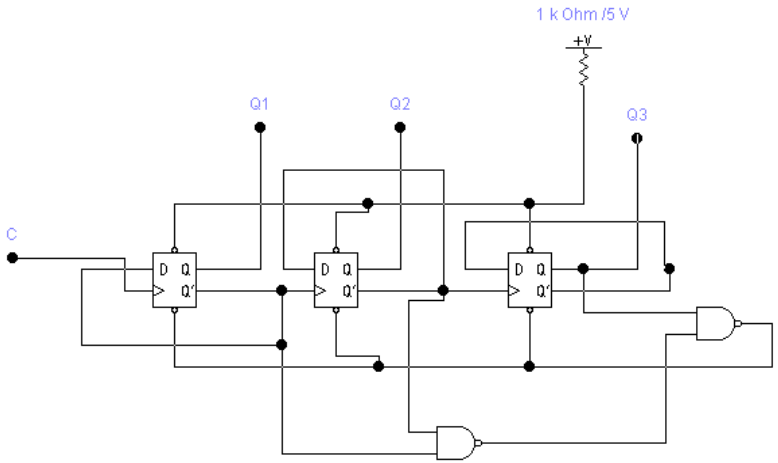


Рис. 4.1. Схема підсумовуючого лічильника

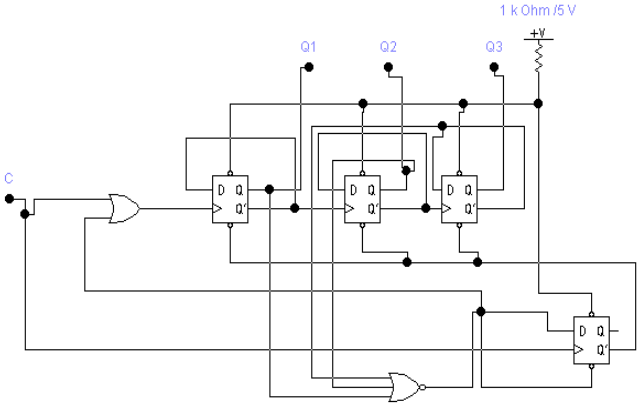


Рис. 4.2. Модернізована схема підсумовуючого лічильника

Експеримент 1. Дослідження підсумовуючого лічильника.

Зберіть схему зображену на рис. 4.3. Включіть схему. Подаючи на вхід схеми тактові імпульси за допомогою ключа *C* і спостерігаючи стан виходів лічильника за допомогою логічних пробників, складіть тимчасові діаграми роботи лічильника, що підсумовує. Визначте коефіцієнт перерахунку лічильника. Результати занесіть в розділ "Результати експериментів". Зверніть увагу на числа, формовані станами інверсних виходів лічильника.

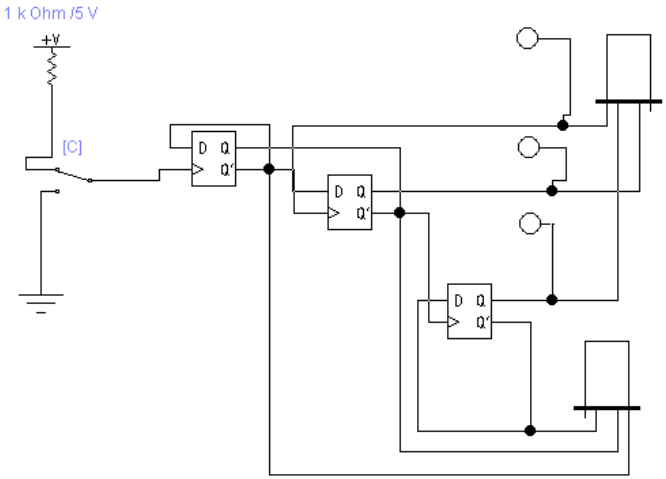


Рис. 4.3. Схема підсумовуючого лічильника

Експеримент 2. Дослідження відіймаючого лічильника.

Зберіть схему зображену на рис. 4.4. Включіть схему. Замалюйте тимчасові діаграми роботи відіймаючого лічильника в розділ "Результати експериментів". В схемі на рис. 4.4 входи логічного аналізатора підключить до інверсних входів тригерів. Включіть схему. Замалюйте одержані тимчасові діаграми в розділ "Результати експериментів" і порівняйте їх з діаграмами, одержаними в попередньому експерименті.

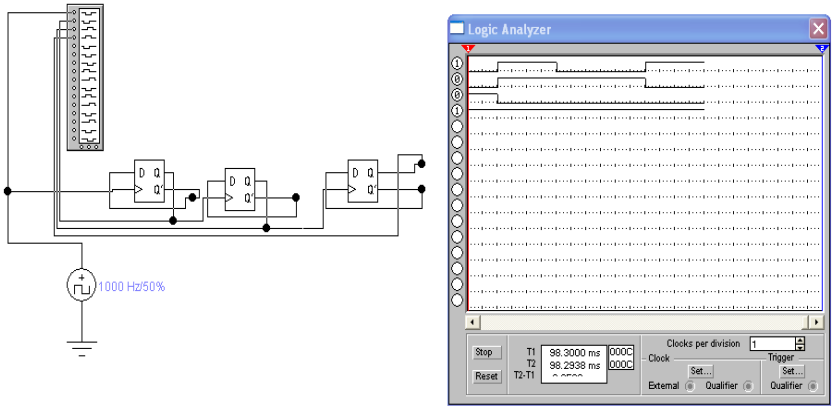


Рис. 4.4. Схема відіймаючого лічильника

Експеримент 3. Дослідження лічильника із зміненим коефіцієнтом перерахунку.

Зберіть схему зображену на рис. 4.5. Включіть схему. Подаючи на вхід схеми тактові імпульси за допомогою ключа *C* і спостерігаючи стан виходів лічильника за допомогою логічних пробників, складіть тимчасові діаграми роботи лічильника і визначте коефіцієнт перерахунку. Результати занесіть в розділ "Результати експериментів".

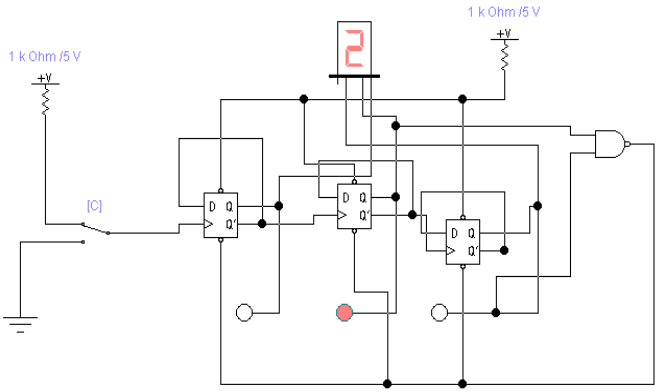


Рис. 4.5. Схема лічильника із зміненим коефіцієнтом перерахунку

Експеримент 4. Дослідження лічильника побудованого на *JK* – тригерах.

Зберіть схему зображену на рис. 4.6. Включіть схему. Лічильник виконаний на трьох *JK* - тригерах в рахунковому режимі (на J- та K- входи подані сигнали 1). Для забезпечення коефіцієнта рахунку $K_{сч}=6$ використаний зворотний зв'язок на елементі **«І»** (*AND_1*), при цьому сигнал 1 з його виходу через елемент **«АБО»** (*OR_1*) поступає на *R* – входи тригерів, переводячи їх в нульовий стан. До другого входу елемента **«АБО»** (*OR_1*) підключений вхід *R* для подачі зовнішнього сигналу скидання.

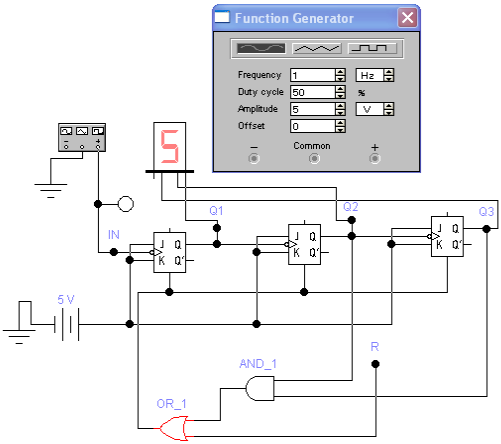


Рис. 4.6. Дослідження лічильника на *JK* – тригерах

Експеримент 5. Дослідження регістра Джонсона, реалізованого на *JK*-тригерах.

Зберіть схему зображену на рис. 4.7. Включіть схему. Встановіть ключ *S* у верхнє положення (на вхід *S* другого тригера подається сигнал логічної одиниці). Побудуйте тимчасові діаграми роботи схеми і занесіть їх в розділ "Результати експериментів". Встановіть схему в стан 000. Подайте за допомогою ключа *S* короткочасний імпульс на вхід *S* другого тригера. При цьому схема

повинна встановитися в стан 010. Подаючи на вхід *C* схеми тактові імпульси за допомогою відповідного ключа і спостерігаючи стан виходів схеми за допомогою логічних пробників, складіть тимчасові діаграми роботи пристрою. Визначте коефіцієнт перерахунку схеми. Результати занесіть в розділ "Результати експериментів".

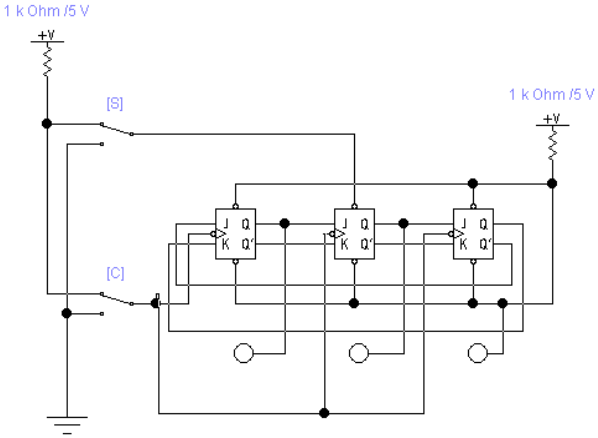


Рис. 4.7. Дослідження регістра Джонсона, реалізованого на JK-тригерах

Хід виконання роботи

1. Повторити конспект (теоретичні відомості).
2. Отримати допуск до роботи.
3. Скласти у *Electronics Workbench* або *Mulsim* відповідні схеми до експериментів.
4. Провести відповідні експерименти. Заповнити протокол та скласти звіт по лабораторній роботі.
5. Зробити висновки по роботі.
6. Відповісти на запитання для самоперевірки.

Запитання для самоперевірки

1. Чому при підключенні рахункових входів тригерів до інверсних виходів попередніх каскадів лічильник на *D*-тригерах працює як що підсумовує, а при підключенні до прямих - як віднімаючий?

2. В якому режимі працюватиме лічильник на *JK*-тригерах при підключенні рахункових входів тригерів до прямих виходів попередніх каскадів? Як зміниться режим роботи лічильника при підключенні рахункових входів тригерів до інверсних виходів?

3. Який коефіцієнт перерахунку має регістр Джонсона?

4. Яким чином можна змінити коефіцієнт перерахунку лічильника?

Лабораторна робота № 5. Експериментальні дослідження дешифраторів та мультиплексорів у програмах Electronics Workbench та Multisim

Мета роботи: Навчитися моделювати роботу дешифраторів та мультиплексорів у програмах Electronics Workbench та Multisim.

Дешифратор - логічна комбінаційна схема, яка має n інформаційних входів і 2^n виходів. Кожній комбінації логічних рівнів на входах відповідатиме активний рівень на одному з 2^n виходів. Звичайно n рівно 2,3 або 4.

На рис. 5.1 зображений дешифратор з $n=3$, активним рівнем є рівень логічного нуля. На входи *C, B, A* можна подати наступні комбінації логічних рівнів: 000, 001, 010...111, всього 8 комбінацій. Схема має 8 виходів, на одному з яких формується низький потенціал, на інших - високий. Номер цього єдиного виходу, на якому формується активний (нульовий) рівень, відповідає числу *N*, визначуваному станом входів *C, B, A* таким чином: $N=C\cdot 2^2 + B\cdot 2^1 + A\cdot 2^0$. Наприклад, якщо на входи подана комбінація логічних рівнів 011, то з восьми виходів мікросхеми (*Y0, Y1...Y7*) на виході з номером $N=3$ встановиться нульовий

рівень сигналу (*Y3=0*), а вся решта виходів матиме рівень логічної одиниці. Цей принцип формування вихідного сигналу можна описати таким чином:

$$Y_i = \begin{cases} 0, & якщо \quad i = k; \\ 1, & якщо \quad i \neq k; \\ k = 2^2 \cdot C + 2^1 \cdot B + 2^0 \cdot A, \end{cases}$$

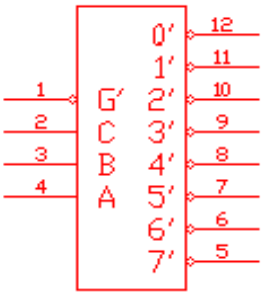


Рис. 5.1. Схема дешифратора з *n=3*

Можна записати вирази для кожного виходу дешифратора:

$$Y0 = \overline{\overline{C} \cdot \overline{B} \cdot \overline{A}}, \qquad Y4 = \overline{C \cdot \overline{B} \cdot \overline{A}},$$

$$Y1 = \overline{\overline{C} \cdot \overline{B} \cdot A}, \qquad Y5 = \overline{C \cdot \overline{B} \cdot A},$$

$$Y2 = \overline{\overline{C} \cdot B \cdot \overline{A}}, \qquad Y6 = \overline{C \cdot B \cdot \overline{A}},$$

$$Y3 = \overline{\overline{C} \cdot B \cdot A}, \qquad Y7 = \overline{C \cdot B \cdot A}.$$

Крім інформаційних входів *A,B,C* дешифратори мають додаткові входи управління *G*. Сигнали на цих входах, наприклад, дозволяють функціонування дешифратора або переводять його в пасивний стан, при якому, незалежно від сигналів на інформаційних входах, на всіх виходах встановиться рівень логічної одиниці. Можна сказати, що існує деяка функція дозволу, значення якої визначається станами управляючих входів. Дозволяючий вхід дешифратора може бути прямим або інверсним. У дешифраторів з прямим дозволяючим входом активним рівнем є

рівень логічної одиниці, у дешифраторів з інверсним входом — рівень логічного нуля. На рис. 5.1 представлений дешифратор з одним інверсним входом управління.

У дешифратора з декількома входами управління функція дозволу, як правило, є логічним множенням всіх дозволяючих сигналів управління. Наприклад, для дешифратора 74138 з одним прямим входом управління *G1* і двома інверсними *G2A* і *G2B* (рис. 5.2) функції виходу *Yi* і дозволу *G* мають вигляд:

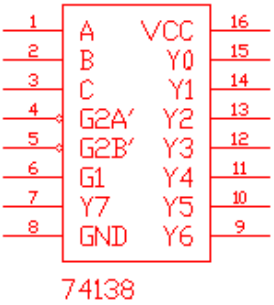


Рис. 5.2. Схема дешифратора 74138

$$Y_i = \begin{cases} \overline{1 \cdot \overline{G}}, & якщо \quad i = k; \\ 1, & якщо \quad i \neq k; \end{cases}$$

$$k = 2^2 \cdot C + 2^1 \cdot B + 2^0 \cdot A,$$

$$G = G1 \cdot \overline{G2A} \cdot \overline{G2B}.$$

Входи управління використовуються для збільшення розрядності дешифраторів або при паралельній роботі декількох схем на загальній вихідній лінії.

Використовування дешифратора як демультиплексора. Дешифратор може бути використаний і як демультиплексор - логічний комутатор, що підключає вхідний сигнал до одного з виходів. В цьому випадку функцію інформаційного входу виконує один з входів дозволу, а стан входів *C, B* і *A* задає номер виходу, на

який передається сигнал з входу дозволу. Порядок проведення експериментів

Експеримент 1. Дослідження принципу роботи дешифратора в основному режимі. Побудуйте схему, зображену на рис. 5.3. Включіть схему. Подайте на вхід *G* рівень логічної одиниці. Для цього клавішею *G* ключ *G* встановити у верхнє положення. Визначте і запишіть рівні сигналів на виходах *Y0...Y7* в таблицю істинності при *G=1* (табл. 5.1 в розділі "Результати експериментів"). Подайте на вхід *G* рівень логічного нуля (ключ *G* встановите в нижнє положення). Переконайтеся, що дешифратор перейшов в робочий режим і на одному з виходів встановився рівень логічного нуля. Подаючи всі можливі комбінації рівнів логічних сигналів на входи *A, B, C* за допомогою однойменних ключів і визначаючи за допомогою логічних пробників рівні логічних сигналів на виході схеми, заповніть таблицю істинності дешифратора при *G=0* (табл. 5.1. в розділі "Результати експериментів").

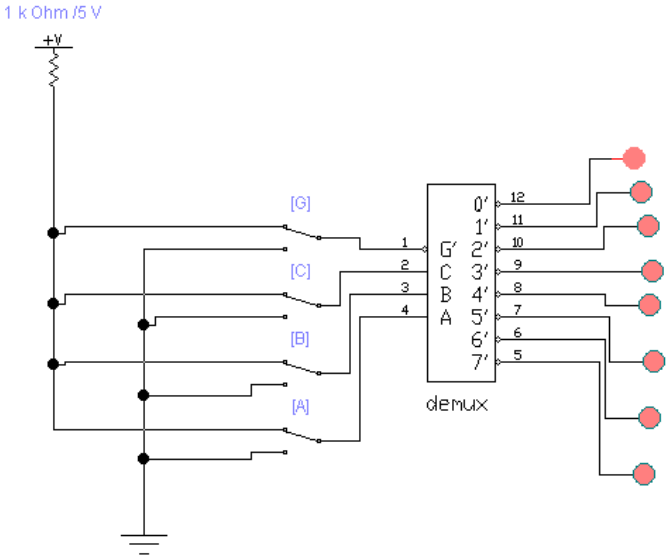


Рис. 5.3. Схема дослідження дешифратора

Таблиця 5.1

Експеримент 1. Результати експериментів.
Дослідження роботи дешифратора в основному режимі

C	*B*	*A*	*G*	*Y0*	*Y1*	*Y2*	*Y3*	*Y4*	*Y5*	*Y6*	*Y7*
0	0	0	1								
0	1	1	1								
0	0	1	0								
0	1	0	0								
0	1	1	0								
1	0	0	0								
1	0	1	0								
1	1	0	0								
1	1	1	0								

Експеримент 2. Дослідження роботи дешифратора як демультиплексора. Побудуйте схему, зображену на рис. 5.4. Включіть схему. У покроковому режимі роботи генератора слів подайте на входи *C, B, A* демультиплексора слова, еквівалентні числам від 0 до 7. Спостерігаючи за допомогою логічних пробників рівні сигналів на виходах, заповніть таблицю функціонування (табл. 5.2 в розділі "Результати експериментів"). Переконайтеся, що сигнал, що змінюється, на вході *G* по черзі з'являється на виходах дешифратора.

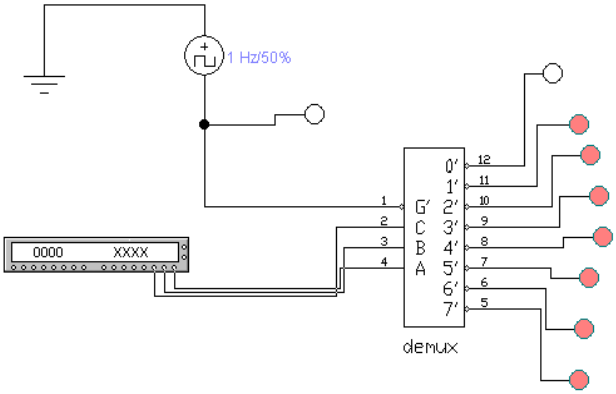


Рис. 5.4. Схема дослідження дешифратора як демультиплексора

Таблиця 5.2

Експеримент 2. Результати експериментів. Дослідження роботи дешифратора як демультиплексора

C	*B*	*A*	*Y0*	*Y1*	*Y2*	*Y3*	*Y4*	*Y5*	*Y6*	*Y7*
0	0	0								
0	0	1								
0	1	0								
0	0	1								
1	0	1								
1	0	1								
1	0	1								
1	0	1								

Мультиплексори.

Мультиплексор - комбінаційна логічна схема, яка керується перемикачем, який підключає до виходу один з інформаційних входів даних. Номер входу, що підключається, рівний числу (адресі), визначуваному комбінацією логічних рівнів на входах управління. Окрім інформаційних і управляючих входів,

схеми мультиплексорів містять вхід дозволу, при подачі на який активного рівня мультиплексор переходить в активний стан. При подачі на вхід дозволу пасивного рівня мультиплексор перейде в пасивний стан, для якого сигнал на виході зберігає постійне значення незалежно від значень інформаційних і управляючих сигналів. Число інформаційних входів у мультиплексорів звичайне 2, 4, 8 або 16. На рис. 5.5 представлений мультиплексор з інверсним входом дозволу *G*, прямим *У* і інверсним *W*-виходами $W = \overline{Y}$.

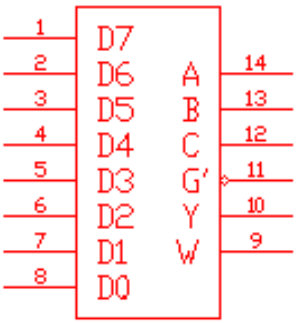


Рис. 5.5. Схема мультиплексора

Функціонування мультиплексора, представленого на рис. 5.5, описується характеристичним рівнянням, що зв'язує сигнал на виході (*У*) з дозволяючим (*G*), вхідними інформаційними (*DO...D7*) і управляючими (*A, B, C*) сигналами:

$$Y = \left(\begin{array}{l} \overline{C} \cdot \overline{B} \cdot \overline{A} \cdot D0 \vee \overline{C} \cdot \overline{B} \cdot A \cdot D1 \vee \overline{C} \cdot B \cdot \overline{A} \cdot D2 \vee \overline{C} \cdot B \cdot A \cdot D3 \vee \\ C \cdot \overline{B} \cdot \overline{A} \cdot D4 \vee C \cdot \overline{B} \cdot A \cdot D5 \vee C \cdot B \cdot \overline{A} \cdot D6 \vee C \cdot B \cdot A \cdot D7 \end{array} \right) \cdot \overline{G}.$$

Логічна функція *n* змінних визначена для 2^n комбінація значень змінних. Кожній комбінації значень аргументів відповідає єдиний інформаційний вхід мультиплексора, на який подається значення функції. Наприклад, вимагається реалізувати функцію

$$F1 = \overline{C} \cdot \overline{B} \cdot \overline{A} \vee C \cdot B \cdot A \vee C \cdot B \cdot \overline{A} \vee \overline{C} \cdot B \cdot A.$$

Ця функція визначена тільки для 8 комбінацій значень змінних, тому для її реалізації пропонується використовувати мультиплексор з трьома управляючими входами. Складемо таблицю істинності функції:

Таблиця 5.3

Таблиця істинності.

N	***C***	***B***	***A***	***F1***
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

З таблиці видно, що для реалізації функції на мультиплексорі необхідно подати на інформаційний вхід мультиплексора з номером *N* сигнал, значення якого рівно відповідному значенню функції *F1*, тобто на входи з номерами *1, 2, 4, 5* слід подати рівень логічного нуля, а на інші - рівень логічної одиниці. Таким чином, при подачі комбінації логічних рівнів на управляючі входи мультиплексора, до його виходу підключиться вхід, значення сигналу на якому рівно відповідному значенню функції. Схемна реалізація приведена на рис. 5.6. При реалізації логічних функцій на інформаційні входи можна подавати не тільки константи, але і вхідні сигнали.

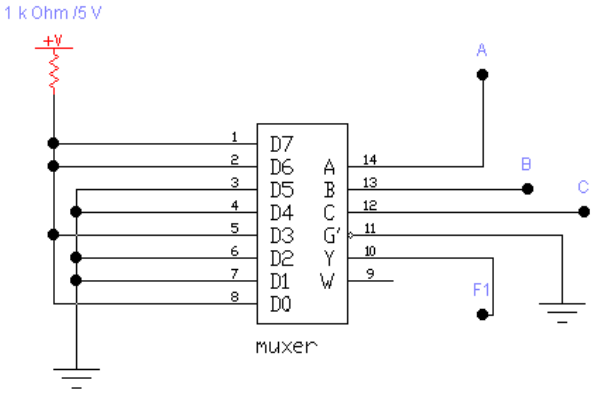


Рис. 5.6. Схема реалізації функцій мультиплексора

Експеримент 3. Дослідження мультиплексора.

Побудуйте схему, зображеною на. рис. 5.7. Включіть схему. За допомогою ключа *G* встановите на вході *G* мультиплексора рівень логічного нуля. По черзі подаючи всі можливі комбінації логічних рівнів за допомогою ключів *A, B, C* на відповідні входи мультиплексора, для кожної комбінації за допомогою логічних пробників визначите, перемикання якого з ключів в лівій частині схеми змінює стан виходів мультиплексора. Позначення відповідного входу мультиплексора запишіть в таблицю 5.4 в розділі "Результати експериментів", вказавши при цьому, як передається вхідний сигнал на виходи мультиплексора (напряму або з інверсією). Наприклад, якщо перемикання ключа 4 змінює стан виходів мультиплексора, в таблиці в рядку з відповідною комбінацією рівнів сигналів на входах *A, B, C* слід записати для виходу *Y - D4*, для виходу *W - D4*. Встановіть за допомогою ключа *G* рівень логічної одиниці на вході *G* мікросхеми. В розділ "Результати експериментів" запишіть позначення висновків, які при перемиканні відповідних ключів в лівій частині схеми не впливають на стан виходів мікросхеми.

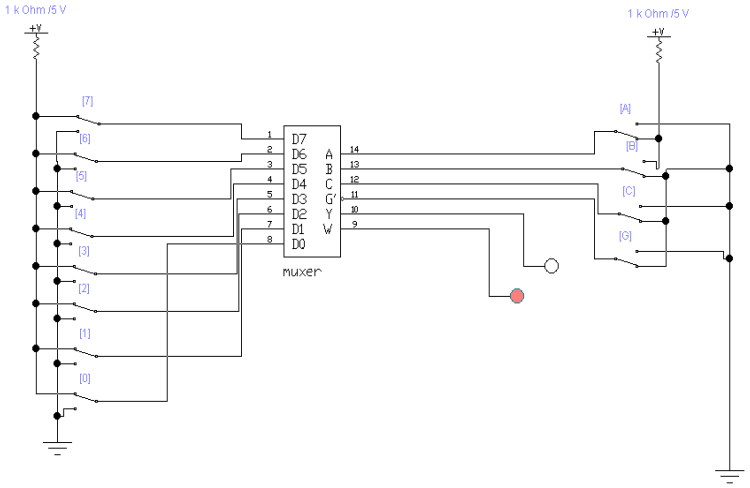


Рис. 5.7. Схема для дослідження мультиплексора

Таблиця 5.4

Експеримент 3. Результати експериментів. Дослідження роботи мультиплексора

A	*B*	*C*	*Y*	*W*
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Хід виконання роботи

1. Повторити конспект (теоретичні відомості).
2. Отримати допуск до роботи.
3. Скласти у *Electronics Workbench* або *Mulsim* відповідні схеми.
4. Провести відповідні експерименти. Заповнити протокол та скласти звіт по лабораторній роботі.
5. Зробити висновки по роботі.
6. Відповісти на запитання для самоперевірки.

Запитання та завдання для самоперевірки

1. Функцію якого електричного пристрою виконує мультиплексор для логічних сигналів?

2. Яким аналітичним рівнянням описується робота мультиплексора з управляючим входом?

3. Як реалізувати схему мультиплексора з управляючим входом на елементах **«І-НЕ»**?

4. Розробіть, зберіть і випробуйте схеми на основі базового дешифратора і елементів «І-НЕ» або, які реалізують задану функцію F. На вході дозволу встановити активний рівень. Варіанти задач приведені нижче.

1) $F = \overline{C} \cdot B \cdot \overline{A} \vee C \cdot B \cdot A \vee \overline{B} \cdot \overline{A}$;

2) $F = \overline{B} \cdot \overline{A} \vee C \cdot \overline{B} \vee \overline{C} \cdot B \cdot A$;

3) $F = \overline{C} \cdot \overline{A} \vee C \cdot \overline{B}$;

4) $F = \overline{C} \cdot \overline{A} \vee B$;

5) $F = \overline{C} \cdot \overline{B} \cdot A \vee C \cdot \overline{B} \cdot \overline{A} \vee \overline{C} \cdot B \cdot \overline{A}$;

6) $F = C \cdot B \cdot A \vee C \cdot B \cdot A \vee C \cdot B \cdot A$;

7) $F = \overline{C} \cdot \overline{B} \cdot \overline{A} \vee B \cdot A \vee C \cdot B$;

8) $F = \overline{C} \cdot B \cdot \overline{A} \vee \overline{B} \cdot A \vee C \cdot B$;

9) $F = \overline{C} \cdot \overline{B} \cdot \overline{A} \vee B \cdot A \vee \overline{C} \cdot \overline{B} \vee C \cdot A$;

10) $F = C \cdot B \cdot \overline{A} \vee B \cdot A \vee C \cdot B \vee \overline{C} \cdot B$;

11) $F = \overline{C} \cdot \overline{B} \cdot \overline{A} \vee C \cdot B \vee C \cdot \overline{A}$;

12) $F = C \vee \overline{B} \cdot A \vee \overline{C} \cdot \overline{A}$.

Лабораторна робота № 6. Моделювання електронних схем комп'ютерів у середовищі MicroCAP-8

Мета роботи: Навчитися аналізувати електронні схеми у MicroCAP-8.

MicroCAP-8 - це універсальний пакет програм схемотехнічного аналізу, призначений для вирішення широкого кола задач. Важливою особливістю цього пакету, є наявність зручного і дружнього графічного інтерфейсу, що робить його особливо привабливим для непрофесійної студентської аудиторії. Не дивлячись на достатньо скромні вимоги до програмно-апаратних засобів ПК (процесор не нижче Pentium II, ОС Windows 95/98/ME або Windows NT 4/2000/XP, пам'ять не менше 64 Мб, монітор не гірше SVGA), його можливості достатньо великі. З його допомогою можна аналізувати не тільки аналогові, але і *цифрові пристрої*. Можливо також і змішане моделювання аналого-цифрових електронних пристроїв

Система MicroCap-8 має в своєму складі один здійснимий модуль - MC8.exe. -  .

MicroCAP-8 є програмою з багато віконним графічним інтерфейсом, що дозволяє будувати і редагувати схеми, моделі і зображення компонентів, а також представляти результати розрахунків в зручному графічному вигляді. Для роботи з цією програмою необхідно використовувати маніпулятор «миша». За допомогою миші можна міняти той, що розташовує і розмір вікон, а також вибирати команди меню. Одночасно можна редагувати декілька схемних файлів, розміщених в різних вікнах.

В інтегрованому пакеті MC8 використовується стандартний багато віконний інтерфейс із спадаючими і розвертаються меню, див. рис. 6.1.

Верхній рядок вікна - **рядок системного меню програми MC8**. На ній зліва знаходиться *кнопка системного меню, що* дублює по зображенню ярлик програми. На цьому ж рядку знаходиться заголовок. Якщо відкрито вікно схем, то указується ім'я файлу схеми і каталогу, в якому він розташований. Якщо ж відкрито вікно аналізу характеристик **Analysis -** указується вид аналізу.

Наступний рядок - **рядок меню команд** програми. Зліва на ній розташована кнопка меню схеми, що управляє розміром вікна схеми, а справа 3 стандартні в WINDOWS кнопки, що виконують аналогічне управління. Між цими кнопками розташовується **спадаюче меню команд програми MC8.**

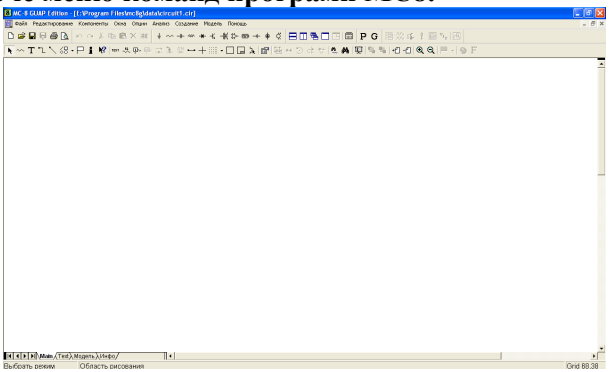


Рис. 6.1. Загальний вигляд програми MC8

Рядок інструментів. На цьому рядку розміщені піктограми самих споживаних команд, всі вони будуть описані нижче при описі відповідних пунктів меню. Піктограми команд негайної дії залишаються натиснутими нетривалий час і потім відновлюють своє первинне положення. Піктограми команд, що перемикають режими, залишаються в положення «включено» до виконання наступної команди.

Лінійки прокрутки. Дві лінійки прокрутки дозволяють панорамувати вікно схем або тексту по горизонталі або вертикалі.

Закладки перемикання вікна схем, вікна тексту. Натиснення на закладку *Text* виводить в робоче вікно зміст текстового вікна, в якому можна розміщувати опис математичних моделей компонентів поточної схеми, директиви і іншу текстову інформацію, або зміст вікна схем. Перемикання між поточним вікном схеми і текстовим вікном проводиться також натисненням гарячих клавіш *Ctrl- G.*

Натискання на закладки Page1, Page2... відкриває відповідну сторінку схеми. Нова сторінка схеми створюється по команді панелі *Add+Page,* що відкривається клацанням правої кнопки миші, коли курсор знаходиться на рядку назви сторінки.

Натиснення на закладку *Models* відкриває текстову сторінку схему, в якій знаходяться модельні директиви, що поміщаються туди по команді меню ***EDIT->Refresh Models***.

При натисненні правої кнопки миші у вікні схем, курсор набуває форму руки, і його переміщення при натиснутій кнопці дозволяє переміщати (панорамувати) схему.

При роботі з MC8 використовується поняття *вибору об'єкту* (компоненту схеми, його позиційного позначення, значення параметра, електричного ланцюга, блоку схеми або рядка тексту). Вибір окремого об'єкту виконується клацанням миші, вибір блоку - висновком його в прямокутну рамку (для цього потрібно клацнути кнопкою миші, помістивши курсор в один з кутів прямокутної області і, не відпускаючи її, розтягнути рамку до необхідних розмірів, після чого відпустити кнопку). Вибраний об'єкт змінює колір; його можна перетягувати за допомогою миші і редагувати.

Для прискорення роботи з програмою використовується не тільки миша, але і клавіатура. Якщо команда меню має підкреслений символ, то ця команда викликається одночасним натисненням клавіш *Alt+підклеслений символ.* Наприклад, меню **Edit** відкривається натисненням клавіш *Alt+E***.**

Команди спадаючого підміню, наприклад **Select All,** викликаються натисненням підкресленого символу, в даному прикладі символу *A*.

Багато команд крім піктограм викликаються натисканням *гарячих клавіш* і комбінацій клавіш. Наприклад, команда видалення з копіюванням в буфер обміну ***Edit->Cut*** викликається натисканням піктограми  , або комбінації клавіш **Ctrl+X.**

Зупинимося докладніше на **спадаючому меню команд програми MC8.**

FILE :

New ( , **CTRL + N**) — створити новий схемний файл (або у форматі MC8 - у вигляді схеми, або у вигляді текстового файлу формату SPICE), або бібліотечний файл, або файл *.MDL моделі для Model.

Open ( , **CTRL + O**) - відкрити для редагування або аналізу схемний файл. Команда викликає діалогове вікно відкриття файлу, за допомогою якого можна відкрити схемний (.CIR, .MAC, .CKT) або бібліотечний файл (.LIB, .LBR, .STM, .MDL, .RES, .CAP, .IND).

Save ( , **CTRL + S**) - збереження схеми з активного з ім'ям і шляхом, вказаним в рядку заголовка.

Save as - збереження схеми з активного вікна в новому файлі, ім'я якого указується в подальшому вікні, що відкривається.

Protect - зберігає файл, що знаходиться в схемному вікні, з паролем в зашифрованому форматі. Після цього вказаний файл можна проглянути або завантажити у вікно схемного редактора тільки ввівши пароль. Захищена схема може бути використана при моделюванні без пароля, якщо вона використовується іншою схемою як макроозначення. Команда використовується для захисту

макромоделей від некомпетентного редагування. Для видалення захисту необхідно зберегти файл командою **Save As**.

Paths - вказівка шляхів розташовує робочих схем (DATA), бібліотечних файлів і файлів (Model Library and Include Files), малюнків, що включаються (picture). Якщо необхідно вказати декілька шляхів, то вони розділяються у відповідному рядку символом «;». MC8 в першу чергу шукає інформацію про моделі компонентів по шляхах, вказаних в рядку даних (DATA) для схем. Потім програма проглядає шляхи, вказані в другому рядку (Model Library and Include Files). Також можна змінити шляхи пошуку моделей, вказавши в самому схемному файлі текстову директиву .PATH.

Cleanup - дозволяє навести лад в робочому каталозі шляхом видалення непотрібних додаткових файлів, що виникають при запуску аналізу. Частіше за все необхідно видаляти вихідні текстові файли з таблицями значень (*.TNO *.ANO *.DNO) і файли даних для графічного постпроцесора PROBE (*.TSA *.ASA *.DSA).

Migrate - дозволяє передати файли компонентів з більш ранньої версії програми MicroCAP. Після того, як буде вказаний той, що розташовує файлу MCAP.DAT старішої версії програми, MC8 прочитає його і сформує список файлів, які можуть бути імпортовані (*.cmp, *.shp, *.pkg).

Translate - перетворення форматів схемних файлів (текстового SPICE в графічний MicroCAP і навпаки, схемного MC8 в схемний більш ранніх версій MicroCAP і ін.).

Binary Library to SPICE Text File: Перетворить бінарний бібліотечний файл формату 1BR в текстовий файл формату .LIB, що містить текстовий опис моделі.

SPICE Text File to Binary Library: Перетворить тестовий бібліотечний SPICE-файл .LIB в бінарний файл програми MC8 формату 1BR.

Schematic to SPICE Text File: Перетворить схемний файл MicroCAP в активному вікні в текстовий схемний SPICE-файл.

Schematic to Printed Circuit Board: Створює файл таблиці з'єднань для подальшого використовування в програмах розводки друкарської платні Protel, Accel, OrCad, або PCADS.

- ***Schematic to Old Version:*** Перетворить схему в активному вікні у формати схем попередніх версій MicroCAP: MC5, MC6, MC7.
- ***Bill Materials:*** Команда створює повний список інформації про компоненти схеми (ім'я, тип, величина, кількість і інші атрибути.
- ***Model to SPICE File:*** Перетворить файл даних Model в текстові модельні повідомлення формату SPICE у вигляді текстового SPICE-файлу.
- ***IBIS to SPICE File:*** Викликає діалогове вікно IBIS, яке створює еквівалентну SPICE-модель для IBIS-файлу (тимчасові діаграми інтерфейсу в спеціальному форматі). Одержаний SPICE-файл можна надалі використовувати при аналізі перехідних процесів.
- ***Touchstone Files:*** Конвертує вихідні параметри S, У, Z, G, Н чотириполюсника в будь-який вказаний формат (S, У, Z, G, Н).

Load MC File - завантаження файлів результатів розрахунку по методу Монте-Карло (*.DNO *.TNO) з подальшим пошуком в ньому варіантів, при яких схема не працює належним чином (установка Report When в MonteCarlo Options). Потім на основі аналізу помилкових варіантів створюються схеми з відповідними параметрами і завантажуються в схемного редактора.

Revert ( , **CTRL+ALT+R**) **-** відновлення вмісту файлу поточного вікна з диска.

Close (CTRL + F4) - завершення роботи з схемою, що знаходиться в активному вікні.

Print Preview () - попередній перегляд зображення перед друком.

Print ( , **CTRL + P**) - висновок на друк зображення в активному вікні відповідно до параметрів, заданих у вікні Print Setup.

Print Setup — вибір принтера і параметрів паперу.

Exit (**ALT + F4**) — завершення роботи з програмою MC8.

EDIT:

Undo ( , **CTRL + Z**) - відміна останньої команди редагування (відкіт назад), залежить від об'єму доступної пам'яті, звичайні не менше 20 дій.

Redo ( ,**CTRL +Y**) - повтор останньої відміненої команди (відкіт вперед).

Cut ( , **CTRL + X**) - видалення обраного () об'єкту і розміщення його в буфері обміну.

Copy ( , **CTRL** + C) - копирование выбранного объекта в буфер обміну.

Paste ( , **CTRL + V**) - копіювання вмісту буфера обміну в поточне вікно в місце на яке показує курсор.

Clear ( , **DEL**) — видалення вибраного об'єкту без копіювання в буфер. **Clear Cut Wire** (**CTRL+DEL**) - обрізання ліній з'єднань (дротів), в точності по межі виділеної прямокутної області.

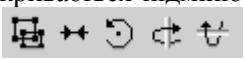
Select all ( , **CTRL + A**) - виділення всіх об'єктів в поточному вікні або всього тексту в текстовому вікні.

Copy to Clipboard - копіювання поточного вікна (або виділеного блоку в ньому) у вигляді різних графічних файлів (відкриваються підміню) в буфер обміну.

Add Page () - додавання до схеми нової сторінки (великі схеми можуть розміщуватися на декількох сторінках).

Delete Page () - видалення однієї (декількох) сторінок схеми.

Refresh Models () - розміщення у вікні Model текстових описів моделей компонентів, які ще небилиці поміщені ні в одне з текстових вікон. Ця команда копіює у вказане вікно інформацію про моделі з бібліотек. Необхідна в тому випадку, якщо необхідно передати схемний файл іншому користувачу, що не має аналогічних бібліотек або для відновлення бібліотечних параметрів моделей в тому випадку, якщо інформація в текстових вікнах редагувалася.

Box - редагування об'єктів, укладених в прямокутну рамку (встановлюється мишею в режимі ). Що відкривається підміню або кнопки піктограм на панелі інструментів  дозволяє копіювати блок вказаного число раз, створювати дзеркально відображений фрагмент, обертати проти годинникової стрілки на 90°, дзеркальне відображати щодо вертикальної і горизонтальної осей, розташованих посередині блоку.

Окремо слід зазначити команду в меню **Make Macro**, що відкривається (**CTRL + M**). Ця команда створює макроозначення з схеми, що міститься усередині прямокутного блоку (дає імена зовнішнім висновкам, зберігає під вибраним ім'ям з розширенням .MAC в каталозі бібліотек, створює компонент в бібліотеці компонентів MACRO.CMP).

Change - зміна ряду параметрів відображення схеми (стає зрозумілим з тих, що відкриваються підміню).

- **Properties (F10):** Відкриває діалогове вікно *Properties* для вікна схеми, в якому можна змінити колір елементів схеми.
- **Graphic Object Properties:** Відкриває діалогове вікно *Graphic Object,* в якому можна змінити властивості відображення графічних об'єктів, таких як лінія, дуга, прямокутник, еліпс і ін. Можна змінити тип лінії, спосіб заливки, кольору. Зміни зачіпатимуть лише нові графічні об'єкти, що додаються редактором. Для зміни властивостей раніше введеного окремого графічного об'єкту необхідно увійти до його властивостей за допомогою подвійного кліка миші.
- **Attributes (CTRL+SHIFT+A):** Дозволяє встановлювати/ скидати відображення 5 основних атрибутів всіх компонентів схеми.
- **Apply Display Properties:** Копіює параметри відображення вибраного компоненту на все інші компоненти даного класу для схеми в активному вікні.
- **Color:** Дозволяє змінити колір вибраного компоненту схеми (електронного компоненту, тексту, сполучного провідника).
- **Font:** Дозволяє змінити параметри шрифту відображення атрибутів вибраного компоненту схеми.
- **Rename Components:** Змінює номери позиційних позначень компонентів згідно загальноприйнятим угодам, так щоб

номери вузлів і компонентів збільшувалися у вибраному напрямі (або зліва направо, або зверху вниз).

- **Rename Defines:** Команда перейменовує символи, визначені директивою Define в тих випадках коли є конфлікт між символьними іменами в різних директивах .Define.
- **Reset Node Positions:** Номери вузлів, потенціали вузлів, струми в гілках і ін. інформація можуть пересуватися мишею по полю схеми з метою легкості для читання. Виконання цієї команди приводить до відновлення прийнятому за умовчанням положенню на схемі вказаної інформації.

Bring to Front () - переміщає нижній об'єкт, що перекривається, вгору.

Send to Back () - переміщає верхній об'єкт, що перекривається, вниз.

Go To Flag. Виконання команди приводить до появи діалогового вікна **Go To Flag.** В ньому можна вибрати потрібну мітку схеми (прапор). При виборі відповідного прапора міняється область схеми, що показується у вікні схемного редактора. Вибір мітки в діалоговому вікні призводить до того, що на екрані відображатиметься частина схеми таким чином, що вибраний прапор знаходитиметься посередині вікна схемного редактора. Прапор ставиться в потрібне місце схеми в режимі встановлення прапора ().

Find ( , **CTRL+F**) - пошук в поточному вікні схем або тексту різноманітних об'єктів.

Repeat Last Find: ( , F3) - повторює команду **Find** з поточними параметрами.

Replace: Проводить заміну тексту в текстовому вікні схеми або в текстовому описі схеми на мові SPICE. Функція заміни текстових атрибутів компоненту схеми виконується установками діалогового вікна Attribute, що викликається за допомогою відповідної команди меню Change.

Find in Files: Команда викликає діалогове вікно *Find in Files,* яке використовується для пошуку на диску файлів з певним змістом. Можна задати, наприклад, фрагмент тексту (TEXT), ім'я (SHAPE), визначення моделі компоненту (DEFINITION), ім'я компоненту

(COMPONENT), величину (ATTRIBUTE). Команда має значення в тих випадках, коли ім'я схемного файлу безнадійно забуто, зате користувач добре пам'ятає який компонент використовувався в ній.

COMPONENT:

Меню COMPONENT містить ієрархічний каталог бібліотек аналогових і цифрових компонентів. Він має систему розвертаються ієрархічних меню, що відкриваються при наведенні на них курсору мишею. В основному назви компонентів зрозумілі з найменування на англійській мові, проте нижче приведений список основних компонентів на російському для кращої орієнтації в програмному пакеті, див. рис. 6.2.

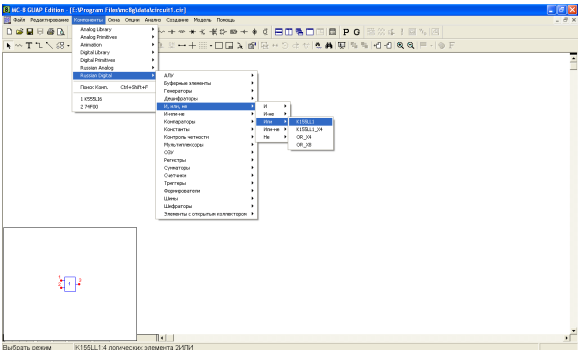


Рис. 6.2. Вибір компоненту для синтезу логічної схеми

Експеримент 1. Дослідження логічних станів електронної схеми у МС-8. Зберіть схему у *МС 8,* див. рис. 6.3. Включіть схему дослідить перехідні процеси у окремому вікні.

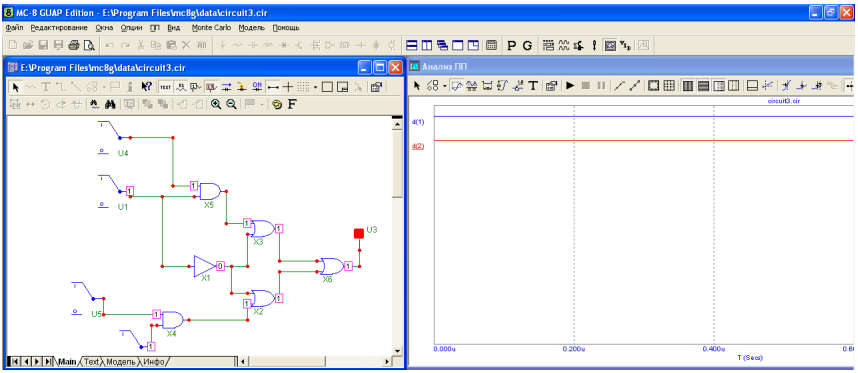


Рис. 6.3. Дослідження логічних станів електронної схеми

Експеримент 2. Дослідження роботи шифратора. Побудуйте схему, зображену на рис. 6.4. Включіть схему. Змінюючи за допомогою перемикачів рівні сигналів на виходах, заповніть таблицю функціонування шифратора.

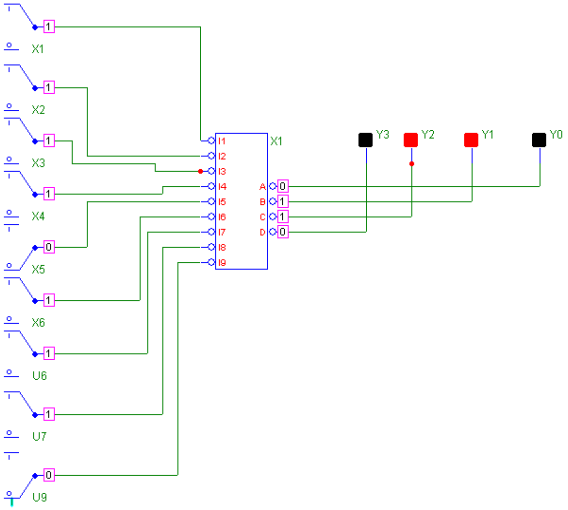


Рис. 6.4. Шифратор на мікросхемі 74147

Експеримент 3. Дослідження роботи дешифратора. Побудуйте схему, зображену на рис. 6.5. Включіть схему. У

покроковому режимі роботи подайте на входи *X1, X2, X3, X4, X5* слова, еквівалентні числам від 0 до 9. Заповніть таблицю функціонування дешифратора.

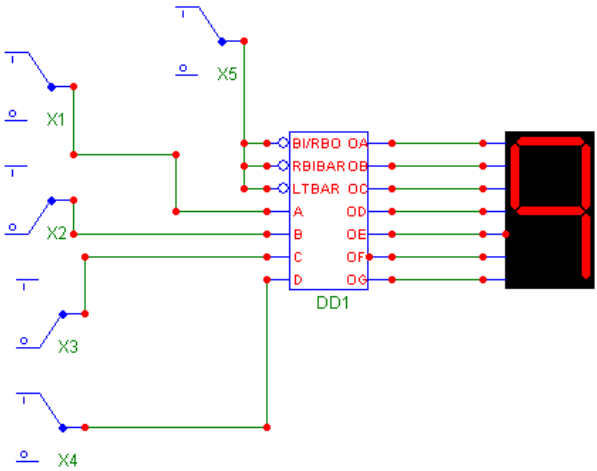


Рис. 6.5. Дешифратор двійкового коду у код семі сегментного індикатора на мікросхемі 7448

Хід виконання роботи

1. Повторити конспект (теоретичні відомості).
2. Скласти у *MC8* відповідні схеми.
3. Провести відповідні експерименти. Заповнити протоколи та скласти звіт по лабораторній роботі. Зробити висновки по роботі.
4. Відповісти на запитання для самоперевірки.

Запитання для самоперевірки

1. За допомогою яких вікон або панелей побудувати комбінаційну схему у *MC8*?
2. Як перевірити працездатність схеми у *MC8*?

3. За допомогою яких інструментів можливо спростити схему у *MC8*?

Лабораторна робота № 7. Моделювання суматорів у Electronics Workbench та MicroCAP- 8

Мета роботи: *Розробити схеми суматорів за допомогою EWB і MC8 та отримати тимчасові діаграми його роботи*

Напівсуматори застосовують лише для складання одно розрядних двійкових кодів або наймолодших значущих розрядів багато розрядних слів. При складанні старших розрядів багато розрядних двійкових слів необхідно в схемі врахувати перенесення з попереднього (молодшого) розряду.

Такий пристрій є повним суматором: він повинен мати три входи (*A, B,* C_{in} – вхід перенесення) і два виходи: *S* – суми, і C_{out} – перенесення.

Складемо логічну функцію повного суматора, див. лабораторну роботу №1:

$$S = \overline{A} \cdot \overline{B} \cdot C_{in} + \overline{A} \cdot B \cdot \overline{C}_{in} + A \cdot \overline{B} \cdot \overline{C}_{in} + A \cdot B \cdot C_{in}.$$
$$C_{out} = \overline{A} \cdot B \cdot C_{in} + A \cdot B \cdot \overline{C}_{in} + A \cdot B \cdot \overline{C}_{in} + A \cdot B \cdot C_{in}.$$

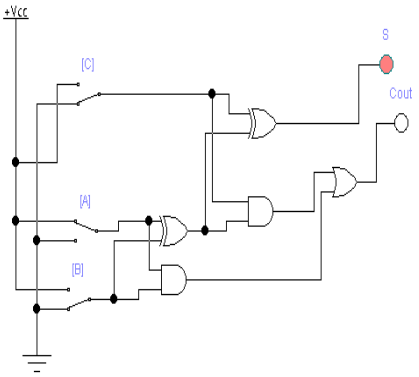
В результаті мінімізації отримаємо наступні вирази

$$S = (A \otimes B) \otimes C_{in} \quad \text{та} \quad C_{out} = A \cdot B + C_{in}(A \otimes B).$$

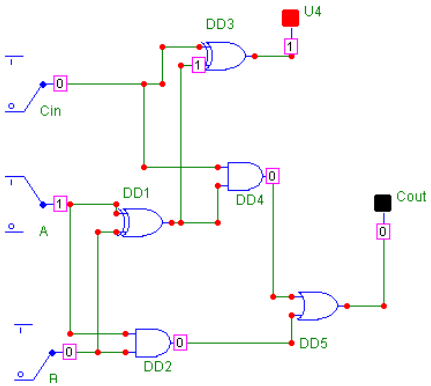
Аналізуючи останні вирази, бачимо, що над *A* і *B* треба провести операцію **XOR** і одержаний результат використовувати двічі: ще раз **XOR** з C_{in} – це дасть *S*, потім операція **AND** з C_{in}, а з її результатом операцію **OR** з $A \cdot B$ (яке виходить як перенесення з першого напівсуматора), і це дасть перенесення C_{out}. Таким чином,

для побудови повного суматора потрібен 5 логічних елементів (ЛЕ) 2 типа **XOR**, 2 типи **AND** і 1 тип **OR**. Виведемо їх на робоче поле і з'єднаємо відповідно до приведених ЛФ (рис. 7.1, а,б). Задаючи різні значення вхідним сигналам, можна спостерігати за роботою суматора. В програмі МС при моделюванні показуються логічні стани всіх цифрових вузлів на схемі, що дозволяє глибше познайомитися з роботою пристрою.

Таким чином, використовуючи два напівсуматори і ЛЕ **OR**, можна побудувати повний суматор.



а)



б)

Рис. 7.1. Логічні схеми повного суматора у EWB та MC8

Суматори на інтегральних мікросхемах

Деякі типи суматорів випускаються в інтегральному виконанні і є одно кристальними пристроями. Як приклад розглянемо роботу КМОП мікросхеми 4008 (чотирьох розрядного повного суматора). Вона складається з чотирьох одно розрядних суматорів і схеми прискореного паралельного перенесення.

В програмі EWB зберемо на основі цієї мікросхеми суматор (див. рис. 7.2). Занесення в даний суматор двох чотирьох розрядних складових $A_3A_2A_1A_0$ і $B_3B_2B_1B_0$ проводиться ключами [1]...[8]. Для зручності відліку двійкових чисел, що вводяться і виводяться, використані семи сегментні індикатори з декодерами (*A, B* і *SUM*).

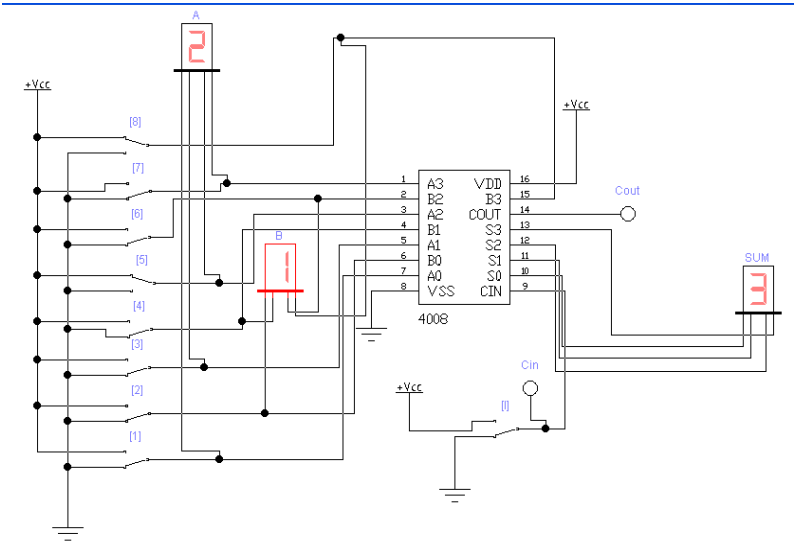


Рис. 7.2. Схема повного суматора на мікросхемі 4008

На вході перенесення з попереднього розряду *Cin* доданий ключ [I]. Цей вхід і вихід перенесення *Cout* контролюються логічними пробниками.

Порада. Перш ніж проводити підсумовування чисел, перевірте роботу схеми, подаючи спочатку тільки на входи

$A_3A_2A_1A_0$ послідовно всі двійкові числа від 0 до 15, потім, аналогічно, на входи $B_3B_2B_1B_0$.

Суматори на мікросхемах можна сполучати один з одним, підключаючи вихід *Cout* попередньої мікросхеми до входу *Cin* – наступної. Так, з двох мікросхем 4008 можна скласти восьми розрядний двійковий суматор.

В програмі MC також розглянемо чотирьохрозрядний повний двійковий суматор, але на основі мікросхем ТТЛ 74283, див. рис. 7.3 та рис. 7.4.

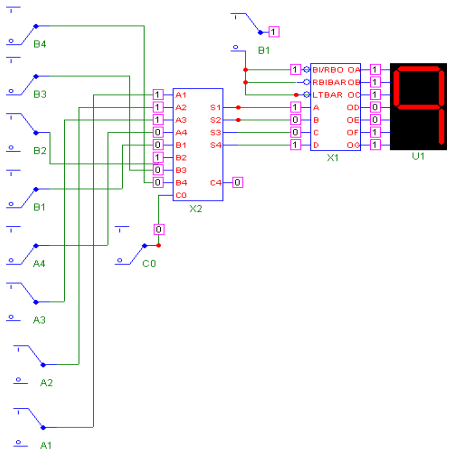


Рис. 7.3. Схема повного суматора на мікросхемі 74283

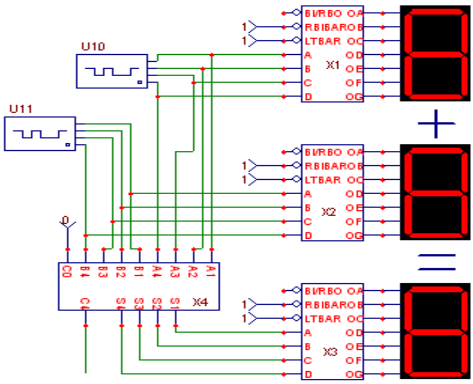


Рис. 7.4. Суматор на мікросхемі 7448

Для налагодження параметрів відповідного елемента, який був попередньо розмішений на екрані, потрібно виділити елемент на екрані, див. рис. 7.4 та двійним кликом лівою клавішею мишки відкрити відповідне вікно, див. рис. 7.5 та задати відповідні режими роботи генераторів.

Для виконання моделювання необхідно потрібним чином запрограмувати тестові послідовності для одно бітових генераторів цифрових сигналів *U10, U11*. одно бітові цифрові генератори вибираються командою *Component->Digital Primitives->Stimulus Generators->Stim1*. При розміщенні генератора на полі схеми (або при його виборі подвійним кліком миші) відкривається вікно завдання його параметрів, в нижній частині якого можна на спеціальній мові задати послідовність логічних станів. До цієї процедури можна порекомендувати розкреслити олівцем на листі клітчастому паперу тестові цифрові сигнали, клітка відповідатиме тривалості одного такту (в нашому прикладі - 1 мкс). Після введення запрограмованої послідовності у вікні завдання параметрів генератора і натиснення ОК текст послідовності з'являється в текстовому вікні Text. Тут вікно схемного редактора розділено на два командою *Windows-> Split Horizontal*. В нижньому вікні показується текстова область схемного редактора.

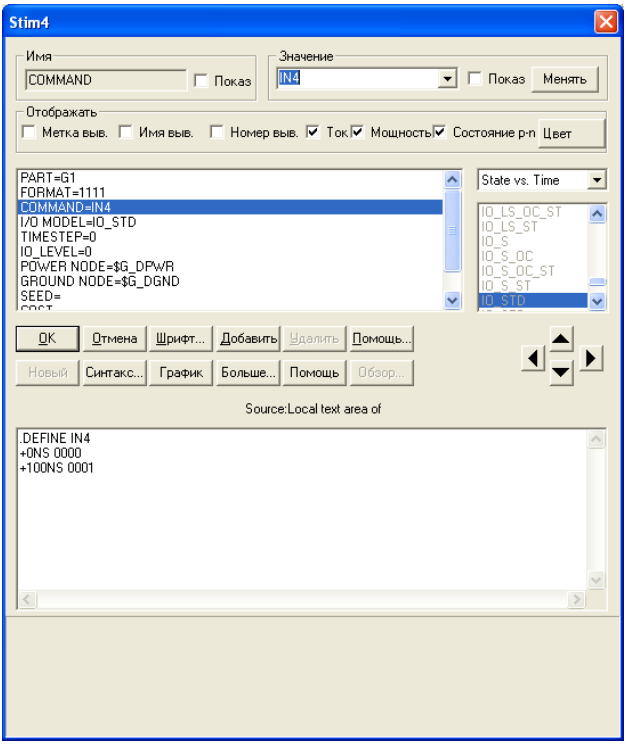


Рис. 7.5. Програмування тестових послідовностей для одно бітових генераторів цифрових сигналів

Для U10	Для U11
.define ina +0ns 0 +100ns 3 +200ns 1 +300ns 0 +400ns 7 +500ns 3 +600ns 5 +700ns 2 +800ns 6 +900ns 3	.define inb +0ns 7 +100ns 6 +200ns 1 +300ns 4 +400ns 1 +500ns 2 +600ns 1 +700ns 0 +800ns 2 +900ns 0

Для виконання аналізу перехідних процесів необхідно вибрати команду меню ***Analysis>Transient*** і заповнити вікно *Analysis Limits*, що відкрилося (рис. 7.6).

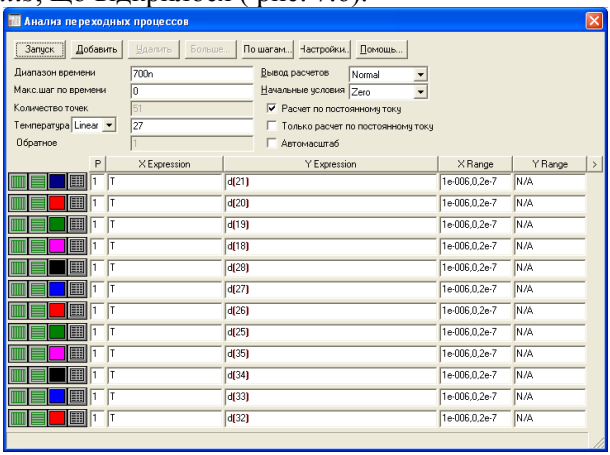


Рис. 7.6. Вікно – «Аналіз перехідних процесів»

Після заповнення необхідних полів залишилося натискувати кнопку RUN і переконатися в правильній роботі синтезованого суматора (рис. 7.7).

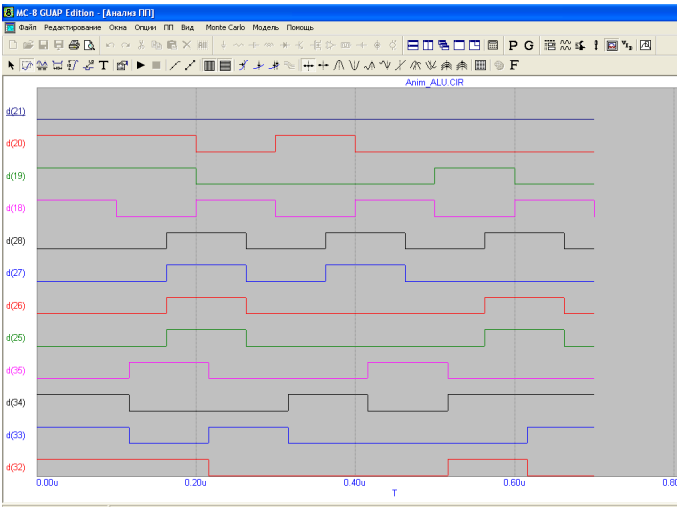


Рис. 7.7. Вікно із тимчасовими діаграмами роботи повного суматора

Хід виконання роботи

1. Повторити конспект (теоретичні відомості).
2. Скласти у *MC8* та *EWB* відповідні схеми повних восьми розрядних суматорів.
3. Провести відповідні експерименти. Заповнити протоколи та скласти звіт по лабораторній роботі. Зробити висновки по роботі.

Запитання для самоперевірки

1. Яким чином отримати схему повного шістнадцяти розрядного суматора у *MC8* та *EWB*?
2. Як перевірити працездатність схеми повного сумматора у *MC8*?
3. За допомогою яких логічних елементів можливо спростити схему у повних суматорів *MC8* та *EWB*?

Лабораторна робота № 8. Моделювання АЛП у Electronics Workbench

Мета роботи: *Ознайомитися із роботою найпростіших АЛП за допомогою EWB та отримати тимчасові діаграми його роботи*

Класична ЕОМ складається з трьох основних пристроїв: арифметико-логічного пристрою, пристрою управління і запам’ятовуючого пристрою. Розглянемо особливості організації цих пристроїв. Перш за все, розглянемо структуру арифметико-логічного пристрою.

Арифметично-логічний пристрій - АЛП, на англійській мові, відповідно, Arithmetic-Logic Unit - ALU, є складною цифровою мікросхемою, яка може виконувати всі комбінаційні функції, а також виконувати ряд інших операцій. Не випадково тому, велика частина інформації в комп'ютерах обробляється саме в АЛП, яке входить складовою частиною в спеціальну інтегральну схему, іменовану центральним процесором (Central Processor Unit – CPU). В первинному комп'ютері IBM PC мікросхема CPU була виготовлена Intel Corporation і названа 8080. АЛП виконує одну з головних функцій мікропроцесора – обробку даних Звичайно АЛП має два вхідні порти і один вихідний. Вхідні порти забезпечуються буферними регістрами для зберігання одного слова даних. Останні в МВ не розглядаються, як, втім, і інші, зв'язані з АЛП складові: акумулятор, регістр станів і шини. Вони разом з АЛП будуть розглянуті в останній роботі, присвяченій моделюванню найпростішої ЕОМ. Перелік функцій власне АЛП залежить від архітектури мікропроцесора і різний для машин різних типів.

В сучасних ЕОМ арифметико-логічний пристрій не є самостійним блоком схемотехніки. Воно входить до складу мікропроцесора, на якому будується комп'ютер. Проте знання структури і принципів роботи АЛУ вельми важливе для розуміння роботи комп'ютера в цілому. Для кращого розуміння цих питань

проведемо синтез арифметичного пристрою, призначеного для виконання тільки однієї операції – додавання чисел з фіксованою комою, заданих в прямому коді, із старших розрядів множника. В ході цього процесу також звернемо увагу на особливості використовування розглянутих вище основних елементів ЕОМ.

Синтез АЛУ проходить у декілька етапів. Спочатку необхідно вибрати метод, по якому передбачається виконання операції, і скласти алгоритм відповідних дій. Виходячи з алгоритму і формату початкових даних, слід визначити набір складових елементів АЛУ. Потім вимагається визначити зв'язки між елементами, встановити порядок функціонування пристрою і тимчасову діаграму управляючих сигналів, які повинні бути подані на АЛУ від пристрою управління.

Хід виконання роботи

1. Повторити конспект (теоретичні відомості).
2. Скласти у *MC8* та *EWB* відповідні схеми АЛУ, див. рис. 8.1 та 8.2.
3. Провести відповідні експерименти. Заповнити протоколи та скласти звіт по лабораторній роботі. Зробити висновки по роботі.

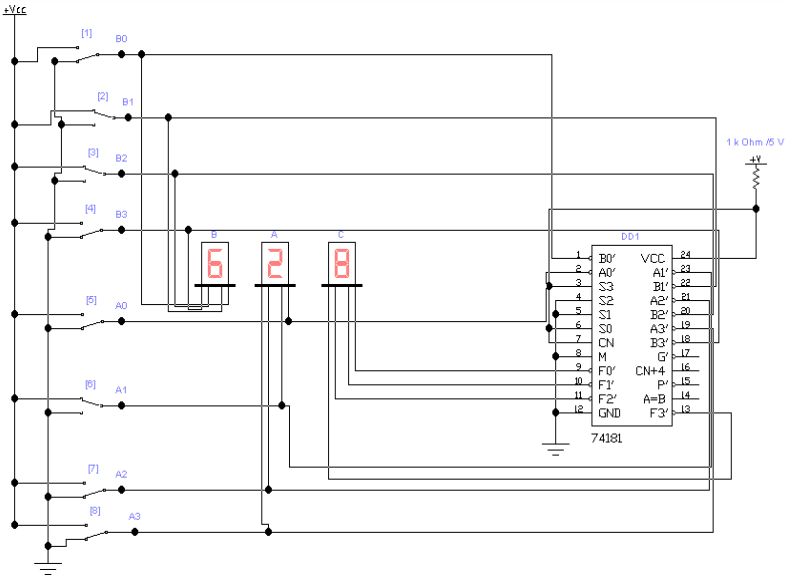


Рис. 8.1. Суматор на чотирьох розрядному АЛУ на мікросхемі 74181

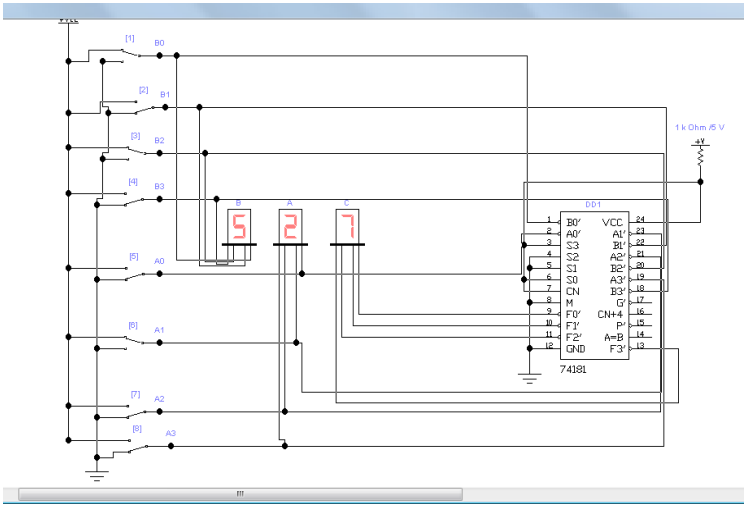


Рис. 8.2. Суматор на чотирьох розрядному АЛУ на мікросхемі 74181

Запитання для самоперевірки

1. У якій послідовності виконується синтез АЛУ?
2. Яким чином отримати тимчасову діаграму роботи складових частин АЛУ?
3. За допомогою яких логічних елементів можливо спростити схему у повних суматорів *MC8* та *EWB*?

Лабораторна робота № 9. Моделювання запам'ятовуючих пристроїв у Electronics Workbench та MicroCAP- 8

Цифрові запам'ятовують пристрої (ЗП), скорочено (пам'ять), використовують для зберігання інформації і обміну нею з іншими пристроями. Це, безумовно, один з найважливіших цифрових пристроїв, визначаючих багато характеристик ПК. Для короткочасного зберігання кодових слів використовують регістри, а для тривалого зберігання і для великих об'ємів інформації служать спеціалізовані мікросхеми. Оскільки мова піде тільки про напівпровідникову пам'ять, то відзначимо, що мікросхеми пам'яті займають до 40% від загального випуску мікросхем.

Найважливішими характеристиками ЗУ є їх інформаційна місткість, організація і швидкодія.

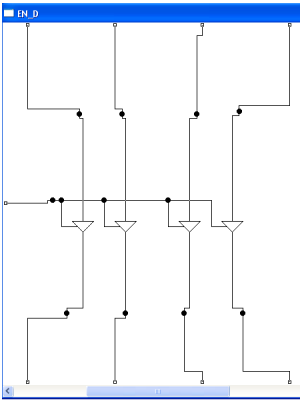
Основною класифікаційною ознакою ЗУ є спосіб доступу до даних (спосіб звернення до масиву елементів пам'яті), по якому вони підрозділяються на адресні, послідовні і асоціативні.

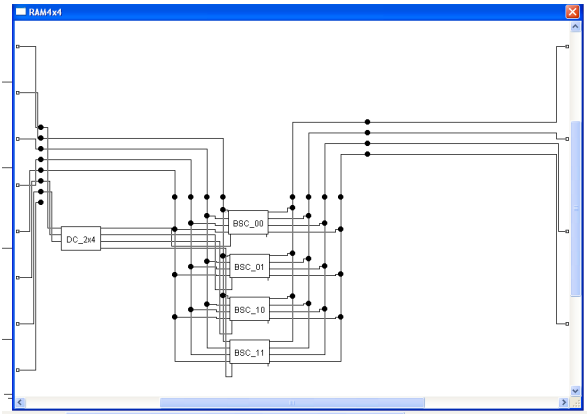
Адресні ЗП у свою чергу підрозділяються на RAM (Random Access Memory - пам'ять з довільним доступом) і ROM (Read-Only Memory - пам'ять тільки для читання). У вітчизняній літературі ці типи ЗУ називаються відповідно оперативним запам'ятовуючим пристроєм (ОЗП) і постійним запам'ятовуючим пристроєм (ПЗП). Перші, як правило, енергонезалежні, а другі, навпаки - енергозалежні. Серед ОЗП розрізняють пристрої статичні - Static RAM (SRAM) і динамічні - Dynamic RAM (DRAM).

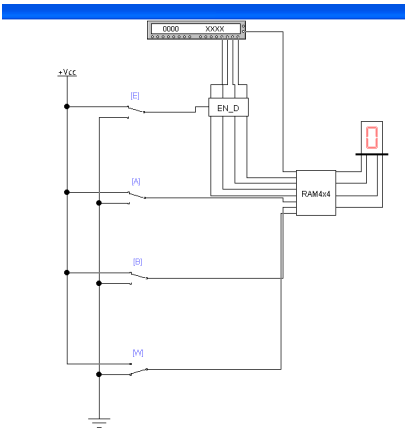
Статичні ПЗП

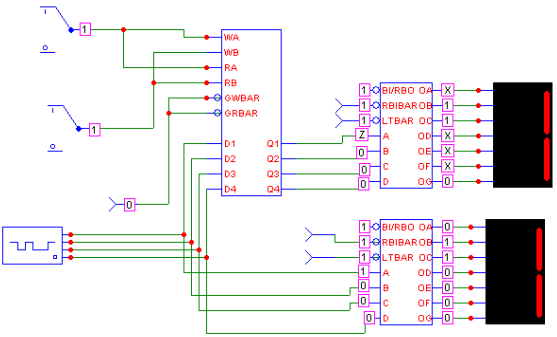
В ЗП з довільним доступом для зберігання кожного біта інформації використовується той, що окремий запам'ятовує, елемент - елемент пам'яті (ЕП). Звичайно, в цій якості використовуються різного роду тригери. Двійкова інформація, занесена в подібний ЗП, зберігається там скільки завгодно довго, поки не буде замінена іншій, або не буде відключено живлення (енергозалежна пам'ять).

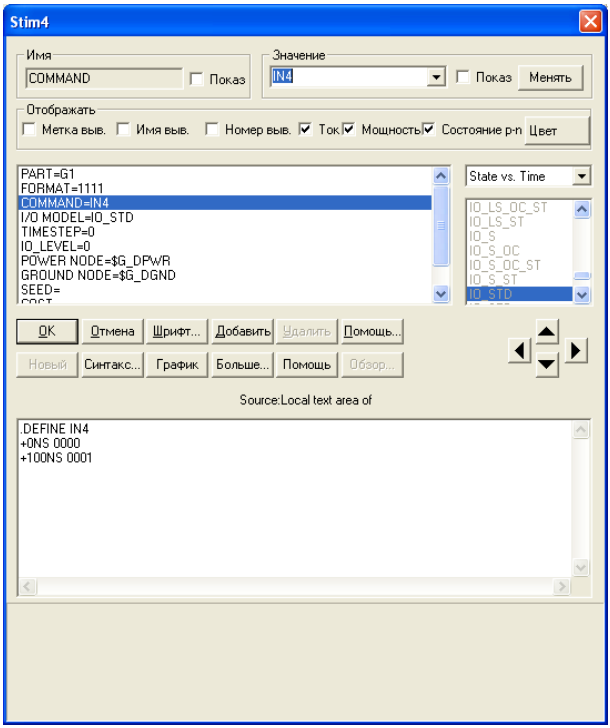
Залежно від способу, яким знаходиться кожен із ЗП в масиві, розрізняють структури з одновимірною (лінійної) і двовимірною адресацією.






0000 XXXX
+Vcc
[E]
EN_D
[A]
RAM4x4
[B]
[M]


WA
WB
RA
RB
GWBAR
GRBAR
D1
D2
D3
D4
Q1
Q2
Q3
Q4
BI/RBO
RBIBAR
LTBAR
A
B
C
D
OA
OB
OC
OD
OE
OF
OG



Хід виконання роботи

1. Повторити конспект (теоретичні відомості).
2. Скласти у *MC8* та *EWB* відповідні схеми.
3. Провести відповідні експерименти. Заповнити протоколи та скласти звіт по лабораторній роботі. Зробити висновки по роботі.

Лабораторна робота № 10. Моделювання простішої ЕОМ у Electronics Workbench

За минулі чотири десятиріччя мікропроцесори пройшли гігантський шлях від першого чипу *Intel 4004*, що містив 2300 транзисторів та виконував близько 60 тисяч операцій в секунду, до сучасних мікропроцесорів *Intel*, *AMD, DEC* і інших провідних компаній, що містять десятки мільйонів транзисторів і виконуючих мільярди операцій в секунду.

Найпростіша ЕОМ, що розглядається на апаратному рівні, складається з п'яти функціональних блоків - пристрій введення, мікропроцесора (включаючи АЛУ і пристрій керування), пам'яті і пристрою виводу. Також, присутній і блок живлення. Зв'язок між окремими блоками здійснюється за допомогою відповідних шин.

Під терміном «шина» розумітимемо сукупність ліній (провідників), по яких передається інформація або енергія від будь-якого з декількох джерел до будь-якого з декількох приймачів.

Через деякі особливості моделюючих програм обмежимося лише розробкою гіпотетичної моделі пристрою для демонстрації принципу дії найпростішої ЕОМ при виконанні деяких функцій.

Побудована модель представлена на рис. 10.1.

В лівій частині схеми (див. рис.10.1, а) розташовуються пристрої введення і управління у вигляді ключів і генератора слів. Далі слідують чотирьох розрядні регістри DD1-DD4 на інтегральних схемах 74175. В правій частині схеми (див. рис.10.1, б) розташовуються: АЛП- DD5 на інтегральній схемі 74181, інвертування - DD6 на інтегральній схемі 7404, синхронний реверсивний програмований лічильник - DD7 на інтегральній схемі 74192.

Чотири мультиплексори - DD8 на інтегральній схемі 74257 і ОЗП.

Вивід проміжної інформації і результату здійснюється на індикатори HG1-HG7. Генератор слів виконує одночасно роль тактового генератора і програмного лічильника.

Розглянемо простий приклад роботи цієї віртуальної моделі ЕОМ. Потрібно скласти два числа 2 і 3.

Виконаємо наступні кроки (команди).

1. Встановимо всі ключі у верхнє положення.

2. Відкриємо генератор слів, встановимо на ньому покроковий режим роботи і задамо перші слова: 0000, 0009, 0002, 0003, 0000, 0000, 0000.

3. Переведемо ключі S в нижнє положення, включимо клавішу моделювання і натискуватимемо на генераторі слів кнопку Step: на шині даних (дисплей HG5) з'явиться цифра 9. Ще раз натискуватимемо Step: на дисплеї HG1 з'явиться цифра 9, на дисплеї HG5 цифра 2.

4. Переведемо ключ S у верхнє положення, ключ A – нижнє положення і натискуватимемо кнопку Step: на дисплеях HG2, HG6 і HG7 з'явиться цифра 2, на дисплеї HG5 – цифра 3.

5. Переведемо ключ A у верхнє положення, ключ B – в нижнє положення і натискуватимемо кнопку Step: на дисплеї Step: HG3 з'явиться цифра 3, на дисплеї HG5 – цифра 1.

6. Переведемо ключ У у верхнє положення, ключ M – в нижнє положення і натискуватимемо кнопку Step: на дисплеях: HG6 і HG7 з'явиться цифра 5.

7. Переведемо ключ З в нижнє положення, натискуватимемо кнопку Step: на дисплеї HG4 з'явиться цифра 0.

8. Переведемо ключ З у Верхнє положення, ключі E і W – в нижнє положенні і натискуватимемо кнопку Step: на дисплеї HG5 з'явиться цифра 5.

Прокоментуємо ці кроки. На кроці 2 програмуються значення, що виводяться на шину даних на подальших кроках на кроці 3 проводиться початкове обнулення регістрів і встановлюється режим роботи АЛЛУ: цифрі 910= 916 = 10012 відповідає режим арифметичного складання операндів при M=0. На кроках 4 і 5 в регістри заносяться операнди: A=2 і B=3. На кроці 6 в АЛЛУ проводяться операція складання. На кроці 7 вибираються осередки ОЗУ з адресою 00 і в них заноситься результат складання Нарешті, на кроці 8 проводиться читання з ОЗУ результату і його висновок на дисплей.

Пораду не вимикайте режим моделювання до закінчення повного циклу (EWB) роботи програми. На початку помилок в

проміжних кроках поверніться до початку і, вимкнувши режим моделювання, наново переустановите ключі.

Хід виконання роботи

1. Повторити конспект (теоретичні відомості).
2. Скласти у *EWB* відповідні схеми.
3. Провести відповідні експерименти. Заповнити протоколи та скласти звіт по лабораторній роботі. Зробити висновки по роботі.

СПИСОК ЛІТЕРАТУРИ

1. Точчи Рональд, Дж., Уидмер, Нил, С. Цифровые системы. Теория и практика, 8-е издание.: Пер. с англ.-М.:Вильямс, 2004.-1024. с.

2. В.І. Жабін, В.В. Т. Цифрові автомати. Практикум. – К.: ВЕК+, 2004. – 160 с., ил.

3. А.Я. Савельев. Прикладная теория цифровых автоматов.- М.: Высшая школа, 1987. - 272 с.

4. К.Г. Самофалов, А.М. Романкевич и др. Прикладная теория цифрових автоматов.-Київ: Вища школа, 1987.-369 с.

5. Ю.Г. Карпов. Теория автоматов.- СПб.: Питер, 2002.-224с.

6. Б.М. Каган. Электронные вычислительные машины и системы. М.: Энергоатомиздат. 1991.- 212 с.

7. А.Г. Филиппов, О.С.Белкин. Проектирование логических узлов ЭВМ. М.: Советское радио.1974.-144 с.

8. Я.М. Будинский. Логические цепи в цифровой технике. М.: Связь. 1977.-164 с.

9. Ч. Гилмор. Введение в микропроцессорную технику. М.: Мир. 1984.-340 с.

10. Шило В.Л. Популярные цифровые микросхемы: справочник, - Москва; металлургия, 1988,-352 с.

11. Цифровые и аналоговые интегральные микросхемы: Справочник / С.В.Якубовский, Л.И.Нильсон, В.И.Кулешова и др./ Под ред. С.В.Якубовского.-М.: Радио и связь, 1990.-496с.