cМІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»
ІНЖЕНЕРНО-ТЕХНІЧНИЙ ФАКУЛЬТЕТ
КАФЕДРА КОМП'ЮТЕРНИХ СИСТЕМ ТА МЕРЕЖ

МЕТОДИЧНІ ВКАЗІВКИ І ЗАВДАННЯ ДО ЛАБОРАТОРНИХ РОБІТ З КУРСУ «КОМП'ЮТЕРНА ЛОГІКА»

для студентів 2-го курсу інженерно-технічного факультету,
напряму підготовки „Комп'ютерна інженерія"

Ужгород – 2016

Методичні вказівки і завдання до лабораторних робіт з курсу „Комп'ютерна логіка" для студентів 2-го курсу інженерно-технічного факультету, напряму підготовки „Комп'ютерна інженерія".

Укладачі: Король І.Ю., канд. фіз.-мат. наук, доцент, завідувач кафедри комп'ютерних систем та мереж;
Тютюнникова Г.С., старший викладач кафедри комп'ютерних систем та мереж.

Рецензент: Бутурлакін О.П.– канд. фіз.-мат. наук, доцент кафедри приладобудування, інженерно-технічного факультету, УжНУ.

Відповідальний за випуск – Туряниця І. І., канд. фіз.-мат. наук, професор, декан інженерно-технічного факультету.

Дані методичні вказівки розглянуто та схвалено на засіданні кафедри комп'ютерних систем та мереж, протокол № 5 від 28 січня 2016 року та методичної комісії інженерно-технічного факультету протокол № 1 від 05 лютого 2016 року.

ВСТУП

Метою викладення дисципліни «Комп'ютерна логіка» є вивчення методів подання чисел в ЕОМ, алгоритмів виконання основних арифметичних та логічних операцій з числами в різних системах числення, основ математичної логіки, аналізу та синтезу цифрових операційних та керуючих автоматів.

Вивчення дисципліни «Комп'ютерна логіка» дає студентам необхідну теоретичну і практичну підготовку для того, щоб вміти розробляти і аналізувати алгоритми перетвореннння дискретної інформації складних процесів, складати структурні схеми комбінаційних логічних схем та автоматів з пам'яттю, ефективно розв'язувати практичні задачі з використанням ЕОМ.

Дані методичні вказівки ознайомлять студентів з маршрутом проектування ПЛІС (Програмовані Логічні Інтегральні Схеми), способами вхідного опису проекту, загальним користувальницьким інтерфейсом системи MAX+plus II (Multiple Array Matrix Programmable Logic User System).

Виконання студентами лабораторних робіт з курсу «Комп'ютерна логіка» дозволить закріпити теоретичні знання і надасть можливість набути практичних навичок роботи із основними редакторами системи MAX+PLUS II.

Лабораторна робота №1

Тема: Початкове знайомство з системою автоматизованого проектування MAX+PLUS II

Мета роботи: ознайомлення з маршрутом проектування ПЛІС (Програмовані Логічні Інтегральні Схеми), способами вхідного опису проекту, загальним користувальницьким інтерфейсом системи MAX+plus II (Multiple Array Matrix Programmable Logic User System) та одержання навичок роботи із основними редакторами системи MAX+PLUS II.

Теоретичні відомості

1. Процес проектування

Традиційний процес проектування ПЛІС складається з наступних етапів: введення проекту, компіляція, верифікація, програмування.

Ведення проекту може здійснюватися декількома способами з використанням: таблиць істинності, булевих функцій (рівнянь), часових діаграм, електричних схем, мов опису апаратури високого рівня, скінченних автоматів, призначенням ніжок, внутрішніх осередків, блоків мікросхеми.

Таблиці істинності та булеві функції використовуються для опису невеликих проектів, оскільки при великій кількості вхідних і внутрішній змінних, табличний опис стає громіздким і незручним, розмірність задачі різко зростає.

Опис за допомогою часових діаграм не користується великою популярністю, тому що не дозволяє оптимально задавати обмеження і вимоги до проектованого пристрою: САПР самостійно вирішує, яким чином реалізувати задані вхідні впливи.

Схемний опис проекту дозволяє за допомогою набору бібліотечних елементів (звичайно це мікросхеми серій 1533, 555 та ін.) і макрофункцій (більше складні елементи, наприклад, RAM, ROM, ALU, порти уведення/виведення і т.д.) задавати алгоритм функціонування проектованого пристрою. Для схемного опису характерна більша вкладеність (ієрархічність) структури: кожен елемент на схемі може представляти окрему електричну схему або текстовий опис (наприклад мовою високого рівня: AHDL, VHDL, Verilog і т.д.). На даний час широке поширення одержує такий метод проектування, при якому блоки пристрою задаються у вигляді елементів, а внутрішній опис елементів − мовою високого рівня (AHDL, VHDL, Verilog і т.д.). Деякі з них (AHDL, VHDL, Verilog) є міжнародними стандартами проектування апаратури. Проект, описаний мовою високого рівня, може без яких-небудь змін і корегувань передаватися між різними пакетами САПР і розроблювачами. Крім того, у багатьох країнах розроблювачі радіоелектронних апаратур і мікросхем зобов'язані поставляти в складі технічної документації моделі на мовах AHDL, VHDL або Verilog. У зв'язку із цим багато фірм-розроблювачів взагалі відмовилися від обов'язкового раніше схемного подання своїх проектів.

Опис за допомогою скінченних автоматів дає можливість наочно й компактно задавати й налагоджувати складні проекти. Практично будь-яка сучасна САПР ПЛІС або система моделювання мають засоби для перетворення автоматного опису в текстовий мовою високого рівня. Налагодження такого проекту полягає в редагуванні структури скінченного автомата, трансляції його в мову високого рівня і моделювання.

Задання алгоритму функціонування пристрою за допомогою призначення логічних блоків ПЛІС використовується рідко і тільки досвідченими розроблювачами. Для цього користуються редактором розведення, в якому в графічному вигляді умовно представлена топологія ПЛІС. Розроблювач вручну вводить логічні зв'язки між логічними блоками для реалізації заданої функції. У цьому випадку може бути отримане оптимальне, з погляду швидкодії й щільності, упакування пристрою. Істотними недоліками такого підходу є висока трудомісткість і неможливість складання документації на розроблений пристрій.

2. Підготовка до виконання лабораторної роботи

Система MAX+PLUS II фірми Altera забезпечує досить широкий спектр можливостей вхідного опису проекту. У її складі є наступні редактори: **Graphic Editor** (графічний редактор), **Symbol Editor** (редактор елементів схем), **Text Editor** (текстовий редактор, підтримуються мови AHDL (Altera Hardware Description Language), VHDL, Verilog та ін.), **Waveform Editor** (редактор часових діаграм), **Floorplan Editor** (редактор розведення). Користувальницький інтерфейс системи MAX+PLUS II при роботі із графічним редактором показаний на рис.1.1.

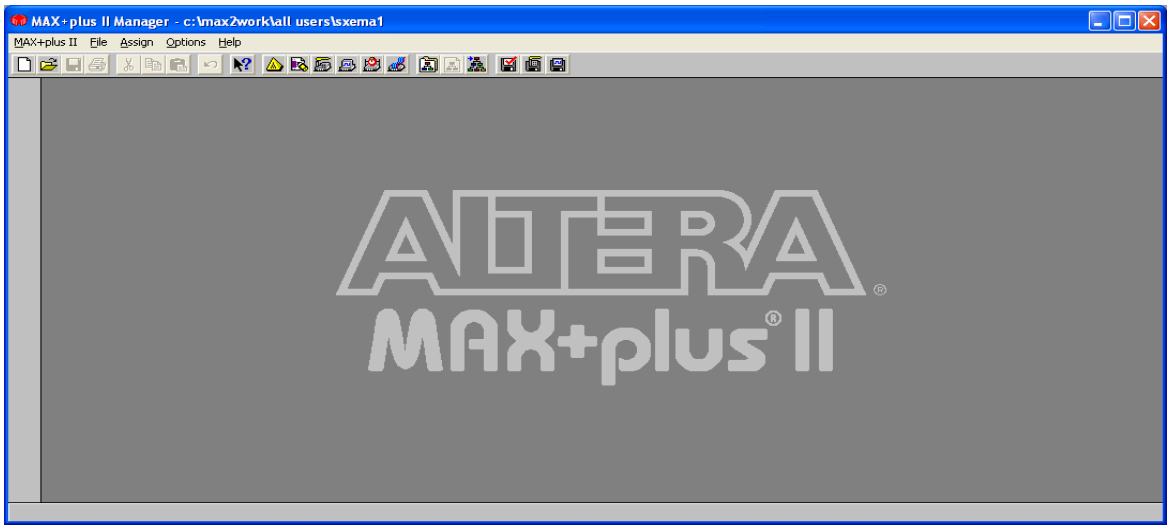


Рисунок 1.1 – Користувальницький інтерфейс системи MAX+PLUS II .

У даній лабораторній роботі розглядається графічний редактор системи MAX+PLUS II (Graphic Editor).

Графічний редактор системи MAX+plus II

Опція **MAX+plus II** основного меню дозволяє відкрити будь-який додаток системи (рис.1.2).

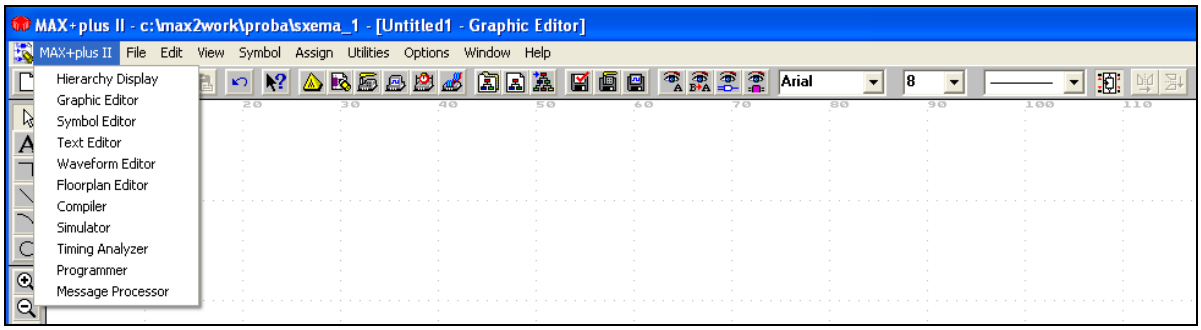


Рисунок 1.2 – Опція MAX+plus II.

Система MAX+PLUS II має потужну вбудовану систему допомоги. Крім того, в основному меню є іконка  , що дозволяє одержувати інформацію про будь-який об'єкт, розташований в активному вікні.

Як правило, сеанс роботи із графічним редактором складається з наступних етапів:

1) створення нового файлу;
2) призначення імені проекту;
3) вибір набору інструментів;
4) уведення логічних елементів;
5) установка розміру та параметрів сітки прив'язки;

6) переміщення символів;
7) уведення вхідних/вихідних контактів;
8) присвоєння імен контактам;
9) з'єднання символів;
10) присвоєння імен ланцюгам і шинам;
11) збереження й перевірка файлу на наявність помилок;
12) створення символу за замовчуванням;
13) закриття файлу.

Розглянемо основні етапи створення проекту з використанням графічного редактора.

Створення нового файлу

Для створення нового файлу слід виконати наступні команди. Вибрати команду **New** (**File menu**). При цьому з'являється діалогове вікно, у якому необхідно вибрати тип файлу - **Graphic Editor File**.

Вибрати тип розширення файлу в меню, що випадає − .gdf. Клацнути на Ok. З'являється вікно графічного редактора без імені - **untitled**'.

Необхідно зберегти файл - команда **Save as**.

Призначення імені проекту

У системі MAX+plus II необхідно призначати робочий файл як поточний проект перед компіляцією, моделюванням й іншими процесами. Система може працювати тільки з одним проектом у даний момент часу. У проект включаються всі файли, що мають однакове ім'я. Розроблювачам **рекомендується** створювати окрему директорію (папку) для кожного нового проекту. Для цього потрібно:

− вибрати команду **Project Name** (**File menu**). З'являється діалогове вікно (рис.1.3);
− задати створений *.gdf - файл як верхній рівень проекту;
− клацнути на **Ok**.

У верхній частині вікна графічного редактора з'являється ім'я проекту.

Як альтернатива команді **Project Name** можна вибрати команду **Project Set Project to Current File** (**File menu**).

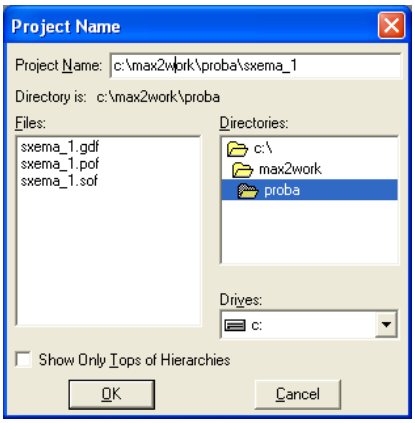


Рисунок 1.3 – Задання імені проекту

Вибір набору інструментів

У лівій частині вікна графічного редактора є меню, що містить набір доступних інструментів **tool**:

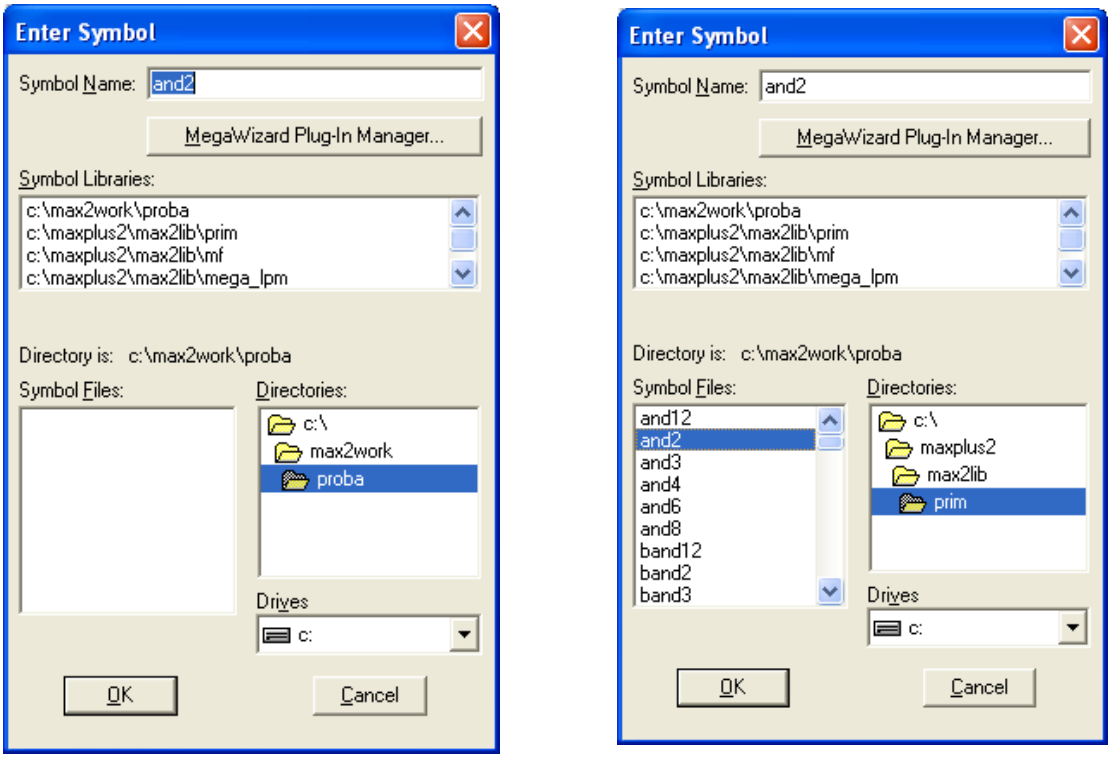
− **selection tool** (інструмент вибору);
− **text tool** (текст);
− **ortogonal line tool** (проведення ортогональних ліній);
− **diagonal line tool** (проведення діагональних ліній);
− **arc tool** (малювання дуг);
− **circle tool** (малювання кіл).

Поточний інструмент вибирається лівою кнопкою миші. При завантаженні редактора за замовчуванням вибирається **selection tool** - інструмент, призначений для вибору об'єктів у вікні та з'єднанні контактів елементів провідниками (має форму стрілки).

Введення логічних елементів

У системі MAX PLUS є бібліотеки, що містять символи різних логічних функцій - примітивні - And, Or, Xor, Not і т.д., мегафункції та макрофункції.

Для введення графічного символу потрібно навести курсор (курсор повинен бути у формі стрілки) на вільну ділянку у вікні графічного редактора й натиснути праву кнопку миші - для визначення точки прив'язки елемента. При цьому поруч із точкою з'являється випливаюче меню (**Symbol menu**). В цьому меню необхідно вибрати команду **Enter Symbol**. На екрані з'являється діалогове вікно **Enter Symbol** (рис.1.4 а)).



а) б)

Рисунок 1.4 − Меню **Enter Symbol**.

Дане вікно дозволяє знайти бібліотечний символ по імені, переглянути вміст всіх підключених бібліотек, вибрати диск і директорію розташування бібліотек.

Наприклад, для введення двовходової логічної функції **I** потрібно в полі **Symbol Name** набрати символ **and2** (рис.1.4 б)). і клацнути на **Ok**. Символ **and2** розміщається верхнім лівим кутом у точці прив'язки. Це ж саме можна зробити клацнувши на бібліотеці примітивів **c:\maxplus2\max2lib\prim**, після чого з'являється список функцій, з якого потрібно вибрити потрібну функцію (уданому випадку функцію **and2).** Далі повторити вищеописані кроки для уведення інших символів.

Якщо логічний елемент є макрофункцією і побудований з використанням більш простих елементів, то його структуру (схему) можна переглянути, навівши на нього курсор і двічі клацнувши ліву кнопку миші.

Якщо потрібно ввести декілька однакових символів, то це можна виконати шляхом копіювання і вставки уже введених символів (див. **Уведення вхідних/вихідних контактів**) або шляхом перетягування символу при натиснутій клавіші **Ctrl**.

Установка сітки прив'язки

Для підвищення наочності й зручності роботи із графічним редактором рекомендується встановити сітку прив'язки символів. Для цього потрібно:

Вибрати **Guideline Spacing** з меню **Options**. З'являється діалогове вікно, після чого надрукувати, наприклад 10, в осередках **X** (**Horizontal**) **Spacing** й **Y** (**Vertical**) **Spacing** і натиснути Ok. Далі вибрати **Show Guidelines** з меню **Options**. У результаті у вікні графічного редактора з'являється сітка з розміром осередку 10 одиниць по горизонталі й вертикалі.

Переміщення та повороти символів

Для переміщення символу потрібно:

Вибрати **Selection tool** з меню, розташованого в лівій частині екрана. Навести курсор на символ. Натиснути ліву кнопку миші й, не відпускаючи її, перемістити символ у необхідну позицію.

Таким чином, у графічному й символьному редакторах можна переміщати будь-які символи, графіку, текстові блоки тощо.

Введені символи або текст можна повертати на 90^0, 180^0 або 270^0. Для цього за допомогою миші виділяємо потрібний елемент або фрагмент схеми і в головному меню лівою клавішею миші клацаємо на опції Edit, після чого на спливаючому вікні вибираємо команду Rotate і потрібний кут повороту.

Введення вхідних/вихідних контактів

Для введення вхідних/вихідних контактів потрібно:

Двічі натиснути ліву кнопку миші у вільній частині екрана, при цьому з'являється діалогове вікно **Enter Symbol**. Надрукувати **input** або **output** у поле **Symbol Name** і натиснути **Ok**. При цьому у вікні з'являється необхідний символ. Якщо схема має кілька входів/виходів, необхідно зробити наступні операції:

− Навести курсор на символ;

− Натиснути клавішу **Ctrl**, ліву кнопку миші й, не відпускаючи їх, переміщати елемент униз;

− Відпустити ліву кнопку миші. При цьому відбулося копіювання елемента. Для продовження копіювання потрібно знову натиснути ліву кнопку миші. Даний спосіб дозволяє копіювати символи без буфера обміну.

Розмножити вже уведені символи можна й традиційним для редакторів типу Microsoft Word методом:

− Навести курсор на необхідний елемент і вибрати його натисканням лівої кнопки миші;

− Натиснути праву кнопку миші, у меню, що з'явився, вибрати команду **Copy** і натиснути ліву кнопку миші. При цьому меню команд зникне;

− Перемістити курсор у місце уведення елемента й натиснути праву кнопку миші. У меню, що з'явився, вибрати команду **Paste** і натиснути ліву кнопку миші.

Кожен уведений символ має ідентифікаційний номер, розташований у лівому нижньому куті границь символу, позначених пунктиром. Номер відповідає порядку, у якому вводилися символи. Крім того, кожному символу за замовчуванням привласнюється ім'я, позначене як **'PIN_NAME'**.

Присвоєння імен контактам

Кожен вхідний/вихідний контакт схеми повинен мати своє ім'я, відмінне від інших. Для цього потрібно:

Навести курсор на поле **PIN_NAME** символу й двічі натиснути ліву кнопку миші.

Надрукувати необхідне ім'я. Якщо після цього натиснути клавішу **Enter**, то автоматично вибереться контакт, розташований нижче для редагування імені.

Уведення імені вхідного/вихідного контакту схеми можна здійснити й з використанням спливаючого меню. Для цього необхідно виділити елемент за допомогою натискання лівої кнопки миші й натиснути праву кнопку миші. У меню, що з'явився, необхідно вибрати команду **Edit Pin Name**.

З'єднання символів

Перед з'єднанням символів (елементів) їх потрібно, по можливості, розмістити так, щоб з'єднати контакти, що перебувають на одному рівні.

Вибрати тонку безперервну лінію в підменю **Line Style** (**Options menu**). Ця лінія рекомендується для побудови провідників.

Натиснути іконку **ortogonal line tool** у лівій частині екрана. При цьому курсор здобуває форму хрестика.

Навести курсор на контакт елемента й натиснути ліву кнопку миші.

Не відпускаючи кнопку, переміщати мишу до наступного контакту або зламу провідника.

Для видалення лінії потрібно повернутися до команди **Selection tool** у лівій частині вікна графічного редактора. При цьому курсор знову прийме вигляд стрілки. Далі необхідно помітити лінію натисканням лівої кнопки миші або виділенням області в, якій вона знаходиться, курсором і натиснути клавішу **Delete** або **Backspace**. Для видалення виділеного фрагмента можна також скористатися спливаючим (після натискання правої кнопки) меню.

Для малювання шин потрібно вибрати товсту безперервну лінію в підменю **Line Style** (**Options menu**).

Присвоєння імен ланцюгам і шинам

Вибрати потрібний розмір шрифту (наприклад вибрати шрифт **Arial** (меню **Option - Font**) розміром -10 (меню **Option -Text Size**)). Тип і розмір шрифту можна також вибрати використовуючи спливаюче (після натискання лівої кнопки миші) меню.

За допомогою подвійного натискання миші позначити необхідний ланцюг. При цьому під ланцюгом з'являється маленька квадратна точка, що показує точку прив'язки імені. Після цього увести ім'я.

Якщо уведене ім'я накладається на символ, його можна перемістити за допомогою миші.

Збереження й перевірка файлу на наявність помилок

Для того, щоб перевірити логічну коректність уведеної схеми, потрібно зберегти файл і перевірити на наявність помилок. Для цього потрібно:

Вибрати **Project Save & Check** з меню **File**. При цьому файл зберігається й на екрані з'являється вікно компілятора (**Compiler window**). Модуль **Compiler Netlist Extractor** перевіряє файл і видає повідомлення про помилки.

Створення символу за замовчуванням

В системі MAX+plus II є можливість створення символу (файл з розширенням ***.sym**), що відповідає уведеній схемі. Створений символ може бути використаний в інших схемах. Для цього потрібно вибрати **Create Default Symbol** з меню **File**. Якщо символ для файлу вже існує, видається запит про можливості перезапису. Більш детально створення символу розглянемо в кінці даної роботи та в наступних лабораторних роботах.

Закриття файлу

Для закриття файлу потрібно натиснути опцію **Close** з меню **File**.

Розглянемо роботу системи MAX+plus II на конкретному прикладі.

Приклад. Користуючись інтегрованим середовищем розробки цифрових пристроїв MAX+plus II спроектувати схему, яка реалізовує логічну функцію, задану таблицею істинності (табл.1.1). Логічна схема такої функції наведена на рис. 1.5.

Таблиця 1.1

A	*B*	*C*	*F*
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

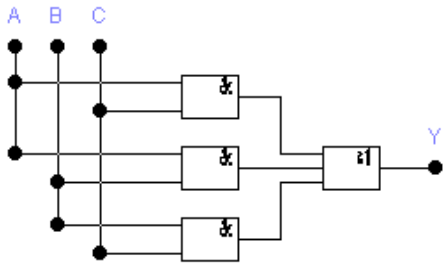


Рисунок 1.5 – Логічна схема

Зауваження. Наведена схема будується на основі відповідної логічної функції, яка одержується наступним чином. За таблицею істинності записуємо функцію Y у вигляді ДДНФ (Досконалої Диз'юнктивної Нормальної Форми):

$$Y = \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C + A \cdot B \cdot \overline{C} + A \cdot B \cdot C,$$

де кожному одиничному значенню функції відповідає конкретний мінтерм.

Далі застосовуючи операцію склеювання $K \cdot \bar{x} + K \cdot x = K$ до кожного із перших трьох доданків і четвертого, маємо:

$$Y = (B \cdot C \cdot \overline{A} + B \cdot C \cdot A) + (A \cdot C \cdot \overline{B} + A \cdot C \cdot B) + (A \cdot B \cdot \overline{C} + A \cdot B \cdot C) = B \cdot C + A \cdot C + A \cdot B.$$

Таким чином ми дістали логічне рівняння, на основі якого будується логічна схема (рис. 1.5).

3. Порядок виконання роботи

3.1 Створення нового файлу та схеми

1. Для створення нового файлу потрібно: в головному меню вибираємо пункт **File**, а в ньому підпункт **New.** При цьому з'явиться діалогове вікно нового файлу (рис.1.6), в якому вибираємо редактор графічних файлів (**Graphic Editor file**) і натискуємо кнопку **OK.**

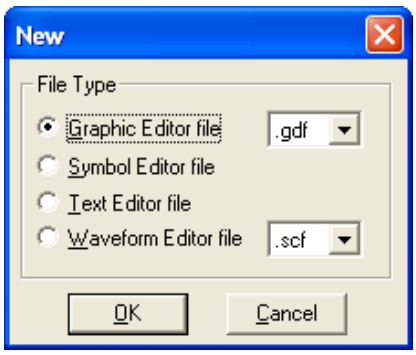


Рисунок 1.6 − Вікно нового файлу

2. Вставляємо елементи, із яких складається схема. Подвійним клацанням лівої клавіші миші у вікні графічного редактора викликаємо вікно введення символу (**Enter Symbol**) (рис. 1.4 а).

3. У вікні введення символу, у списку бібліотек символів (**Symbol Libraries**), вибрати бібліотеку примітивів (**prim**) і натиснути кнопку **OK.** Після цього в списку (**Symbol Files)** появляться усі елементи даної бібліотеки (рис. 1.4 б).

4. Вибрати потрібний елемент і натиснути кнопку **OK.** Даний пункт повторити для всіх елементів.

Примітка. *Входи і виходи логічних елементів називаються виводами, а входи і виходи схеми − контактами*.

5. Вставити вхідні і вихідні контакти. Для цього в списку (**Symbol Files)** вибрати (**input**) для вхідних контактів і (**output**) для вихідних.

6. Зв'язати елементи з'єднувальними лініями (провідниками). Для цього підвести курсор миші до виводу елемента (при цьому курсор змінить вигляд на перехрестя), натиснути ліву клавішу миші і, не відпускаючи її, вести мишу до потрібного виводу потрібного елемента.

7. Дати імена усім вхідним і вихідним контактам схеми. Для цього підвести курсор миші до (**Pin_Name**), натиснути праву кнопку миші і у виплившому контекстному меню вибрати пункт (**Edit Pin Name**). Імена повинні бути унікальні!

8. Зберегти створений файл під іменем **sxema1.gdf**. Для цього в меню вибрати (**File→Save As)** і в діалоговому вікні в папку (наприклад, **lab_1)** ввести ім'я файлу і натиснути кнопку **OK.** При цьому появиться запит (рис.1.7), який треба підтвердити, якщо, звичайно, все правильно.

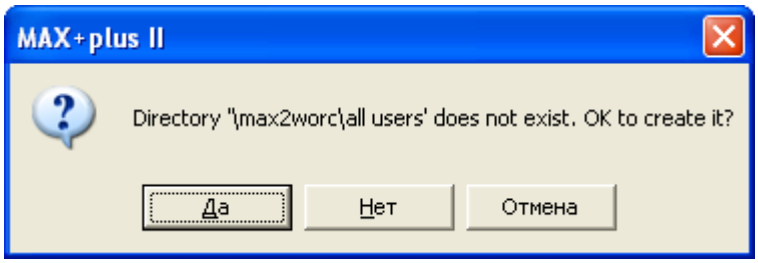


Рисунок 1.7 – Запит на створення директорії.

Схему проекту, виконану в графічному редакторі, наведено на рис. 1.8.

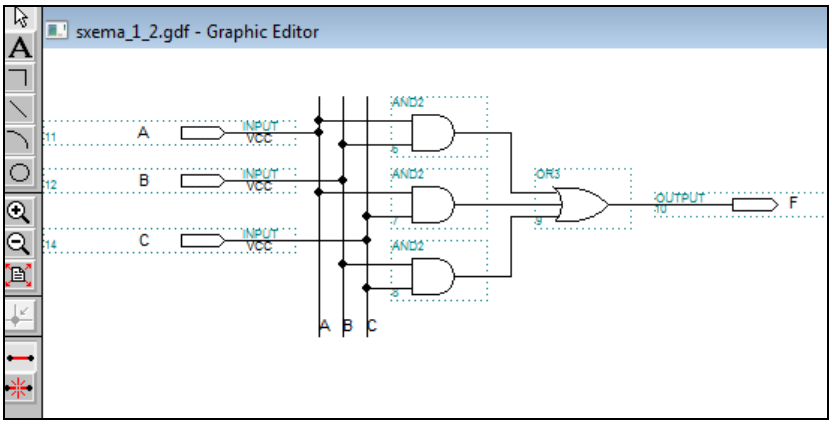


Рисунок 1.8 − Схема проекту, виконана в графічному редакторі

3.2 Процес компіляції

1. Перед компіляцією потрібно вибрати сімейство і підходящий тип мікросхеми, на якій буде реалізована схема. Для цього в головному меню потрібно вибрати пункт **Assign→Device.** При цьому появиться вікно **Device** (рис.1.9)

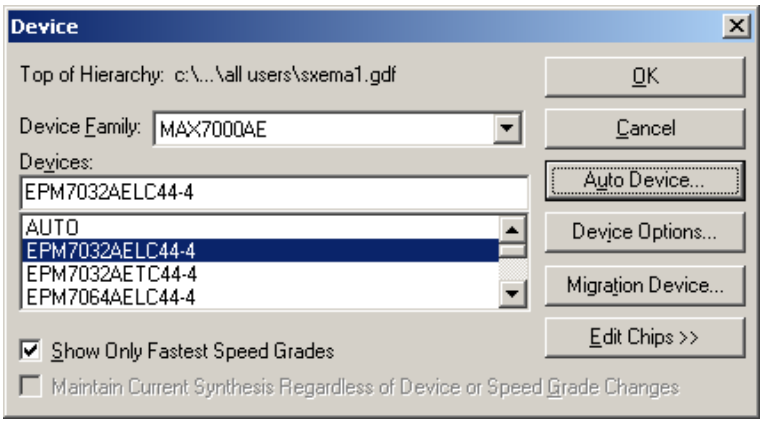


Рисунок 1.9 – Вікно вибору мікросхеми

2. В головному меню вибрати пункт **File→Project →Save&Compile**. Якщо схема ще не була збережена, то потрібно її зберегти, використавши вікно Project Name (рис.1.10) і відкомпілювати. Якщо помилок немає, то появиться вікно компілятора з результатами компіляції (рис. 1.11). В процесі компіляції можуть бути виявлені помилки компіляції (**Error**), видані попередження (**Warning**) і повідомлення (**Info**). Усі вони виводяться у вікні (**Message-Compiler**).

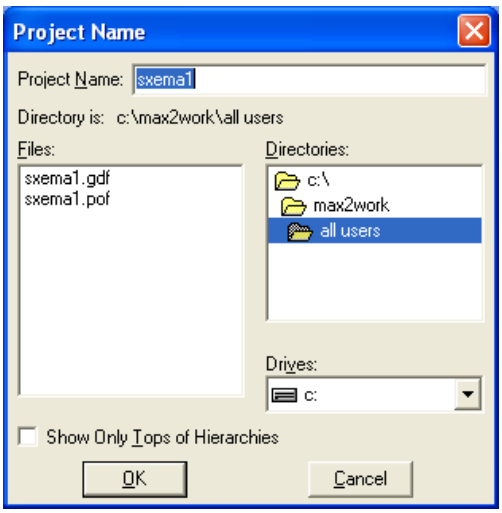


Рисунок 1.10 − Задання імені проекту

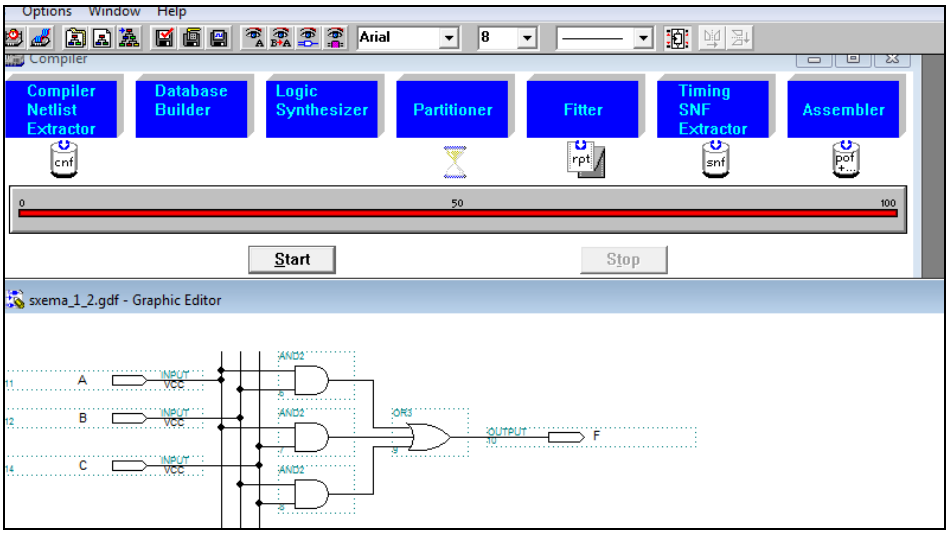


Рисунок 1.11 − Вікно компілятора з результатами компіляції

3. Після успішної компіляції проекту можна подивитись і відредагувати розміщення виводів на рисунку корпусу пристрою. Для цього в головному меню потрібно вибрати пункт (**MAX+plus II→Floorplan Editor**) (рис.1.12).

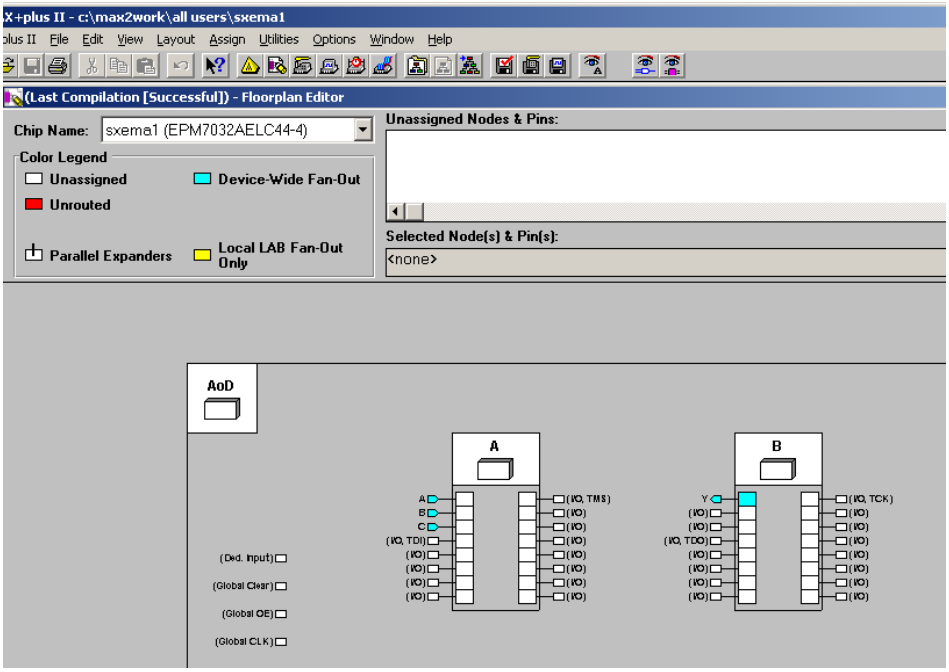


Рисунок 1.12 – Схема корпусу пристрою

3.3. Моделювання

Моделювання дозволяє тестувати логічні функції (операції) й внутрішню синхронізацію проектованої логічної схеми. Моделювання можна проводити тільки після успішної компіляції. Сигнальний редактор виконує дві ролі: служить інструментом створення дизайну й інструментом введення тестових векторів і перегляду результатів тестування. Моделювання складається з наступних кроків.

1. Створити сигнальний файл. У головному меню вибрати пункт **File**, а в ньому підпункт **New**. З'явиться вікно діалогу вибору типу нового файлу (рис.1.6):

2. Вибрати редактор сигнальних файлів (**Waveform Editor file**) і натиснути кнопку **OK**. З'явиться вікно сигнального редактора (рис.1.13)

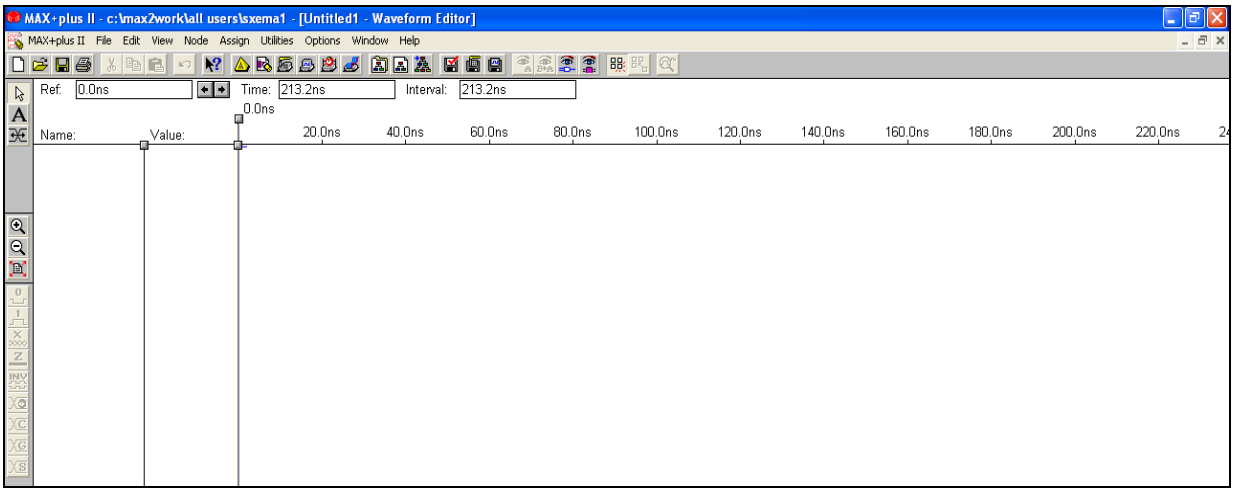
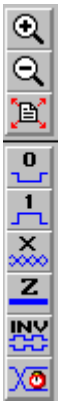


Рисунок 1.13 − Вікно сигнального редактора

Основні інструменти сигнального редактора:



Збільшити масштаб
Зменшити масштаб
Зменшити масштаб так, щоб вся діаграма вмістилася на екрані
Установити низький логічний рівень(0)
Установити високий логічний рівень(1)
Установити невизначений логічний рівень(X)
Стан високого імпедансу (Z-стан)
Інвертувати логічні рівні
Задання змінного сигналу з постійною частотою.
Крім того, вище лінійки інструментів, біла похила стрілка означає установку репера (що пересуває по часовій шкалі вертикальну оцінку); ↔ означає можливість варіювання положенням фронтів (тривалістю) сигналу.

3. У вікні сигнального редактора потрібно вставити контакти, на яких буде вироблятися моделювання сигналів. Вибрати пункт **Node** головного меню, а в ньому підпункт **Enter Node from SNF**. З'явиться вікно **Enter Nodes from SNF** (рис.1.14)

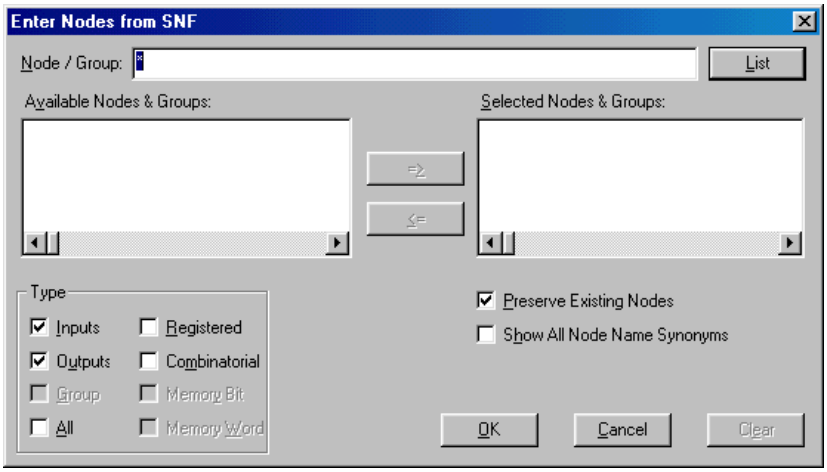


Рисунок 1.14 – Вікно вибору контактів

4. У вікні **Enter Node from SNF** натиснути кнопку **List**. У лівому списку (**Available Nodes & Groups**) з'являться всі доступні для моделювання контакти. Виділені в лівому списку контакти можна вибрати за допомогою кнопки "⇒" і натиснення **OK**. Очистити список вибраних контактів (**Selected Nodes & Groups**) можна за допомогою кнопки **Clear** (рис. 1.15).

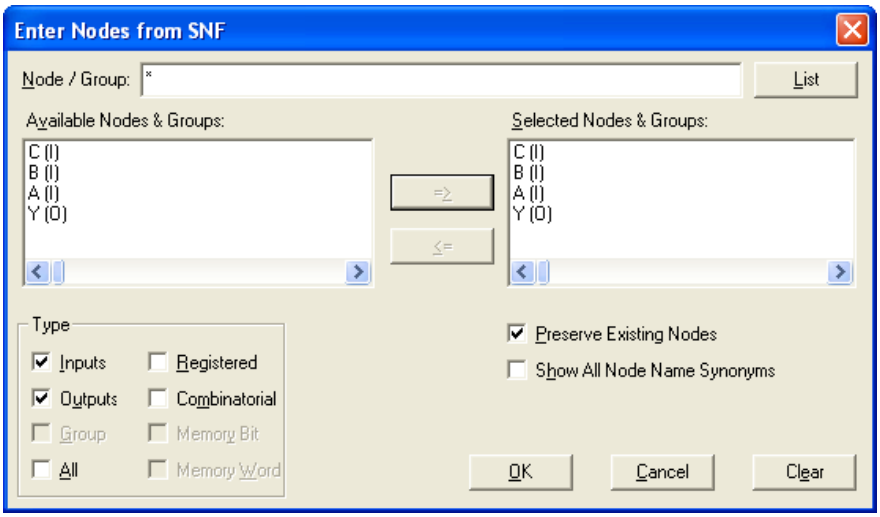


Рисунок 1.15 – Вікно вибору контактів зі списком доступних контактів

5. Задати активні вхідні впливи (одиниці вхідного коду). Для цього виділити за допомогою миші частину часової діаграми контакту, на який подається логічна одиниця (логічний нуль), і натиснути кнопку інструментів “1 - установити високий логічний рівень” (“0 – установити низький логічний рівень”). У результаті на виділених контактах у відповідний час буде логічна одиниця. За допомогою кнопок інструментів установити початкові значення вхідних сигналів, масштаб часу порядку 1 поділка - 10,0 ns. Натиснути кнопку-інструмент ↔ , появиться курсор у вигляді перехрестя, пересуваючи його, натиснувши ліву кнопку миші, задати часові діаграми вхідних сигналів у формі прямокутних імпульсів різної довжини, зачорнюючи обраний відрізок часу й використовуючи кнопки установок рівнів логічних "0" й "1". Один із вхідних сигналів сформувати як меандр, використовуючи вкладку натисканням правої кнопки миші: Insert Node ..., Enter nodes from SNF. Установити кратність періоду меандру Multiplier: 1,2,4,8 і т.д. за допомогою інструменту сигнального редактора задання змінного сигналу з постійною частотою.

6. Для виконання моделювання у головному меню потрібно вибрати **File→Project →Save &Simulate**. Натиснути **Start**. Відкрити графічний редактор з результатами моделювання (кнопка **Open SCF** цього вікна). З'являться сигнали на вихідних контактах - результат роботи пристрою (рис.1.16). Вихідні сигнали з'являться із затримкою, що залежить від швидкодії обраної мікросхеми й складності проектованої схеми. Результати моделювання можна перевірити і в числовому вигляді, вставивши вертикальну риску на відповідну комбінацію (набір) вхідних даних (на рис. 1.16 − це набір 110).

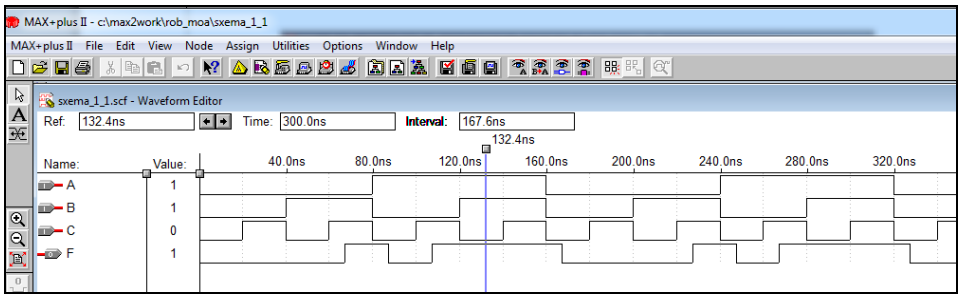


Рисунок 1.16 – Результати моделювання

Аналіз часових затримок

Для аналізу часових затримок запускаємо часовий аналізатор **Timing Analyzer** і натискуємо кнопку **Start**. Результати часового аналізу наведено на рис.1.17. На даному рисунку видно, що затримка проходження сигналу від входів *A, B* і *C* до виходу Y 6,0 ns.

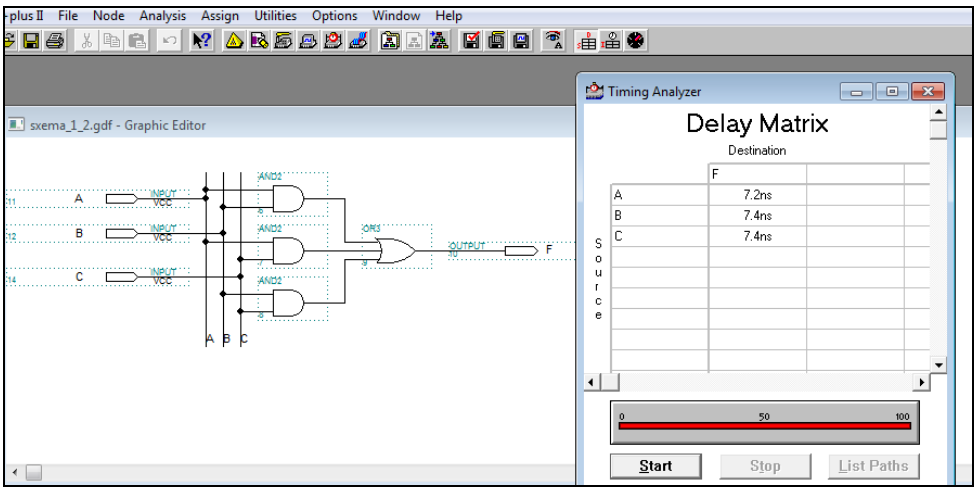


Рисунок 1.17 − Часовий аналізатор затримок

Створення бібліотеки

В системі **MAX+plus II** є можливість включити свою бібліотеку в список стандартних бібліотек символів.

1. Вибрати пункт **Options** головного меню, а в ньому підпункт **User Libraries**. З'явиться вікно **User Libraries**

2. У списку **Directories** вибрати свою директорію, натиснути кнопку **Add,** а потім кнопку **OK**.

Створення символу

Символ - це логічна схема, представлена у вигляді одного елемента. Символьний файл має те ж ім'я, що й файл проекту і розширення **.sym**. Для створення символу потрібно:

1. Перебуваючи у вікні графічного редактора виберіть у головному меню пункт **File**, а в ньому підпункт **Create Default Symbol**, буде створений символ.

2. Щоб подивитися або змінити символ, виберіть у головному меню пункт **File**, а в ньому підпункт **Edit Symbol**, – з'явиться вікно символьного редактора.

3. Створений символ можна вставити в іншу схему. Подвійним клацанням лівої кнопки миші у вікні графічного редактора викликаємо вікно уведення символу (**Enter Symbol**).

4. У вікні уведення символу, у списку бібліотек символів (**Symbol Libraries**) вибрати свою бібліотеку й натиснути кнопку **OK**, після цього в списку **Symbol Files** з'являться всі елементи даної бібліотеки.

5. Вибрати потрібний елемент зі списку **Symbol Files** і натиснути **OK**. Обраний елемент з'явиться у вікні графічного редактора.

Перегляд файлів проекту

Переглянути результат проектування - всі створені файли в **Max+plus→Hierarchy Display:**

gdf - вашу принципову схему,

tdf - опис цієї схеми мовою програмування,

rpt - інформацію про ваш проект: корпус обраної вами раніше мікросхеми із вказівкою зайнятих вашою схемою контактів,

scf - часові діаграми функціонування схеми,

pin - номери й назви контактів обраної мікросхеми, у якій розміщена проектована схема і її технічні характеристики

fit - номери й назви тільки тих контактів мікросхеми, які зайняті проектованою схемою,

acf - опис усього проекту: опцій, компіляції, симуляції, тимчасового аналізатора й ін.

Лабораторна робота №2

Тема: Дослідження булевих функцій двох змінних

Мета роботи: Ознайомитися з основними логічними функціями (операціями). Оволодіти навичками роботи в системі проектування MAX+plus II для моделювання роботи простих цифрових схем.

Теоретичні відомості

1. Основні поняття та визначення булевої алгебри

Математичний апарат, який описує дії дискретних пристроїв, базується на алгебрі логіки, її ще називають по імені автора – англійського математика Джорджа Буля (1815 – 1864) булевою алгеброю. Практичне застосування алгебри логіки першим знайшов американський вчений Клод Шеннон у 1938 р. при дослідженні електричних кіл з контактними вимикачами.

Для формального опису цифрових автоматів використовується апарат алгебри логіки. Логічною (булевою) змінною називається величина, яка може приймати тільки два значення 0 і 1. Сукупність різних значень змінних називаються набором.

Основним предметом булевої алгебри є висловлювання – просте твердження, про яке можна стверджувати: істинне воно (позначають символом 1) або хибне (позначають символом 0). Прості висловлювання позначають буквами, наприклад $x_1, x_2, \ldots, x_n$, які у цифровій техніці називають змінними (аргументами).

У даний час головна задача алгебри логіки — аналіз, синтез і структурне моделювання будь-яких дискретних скінчених систем.

Логічну змінну із скінченим числом значень (станів) називають перемикальною, а зі двома значеннями — булевою.

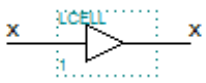
Операція — це чітко визначена дія над одним або декількома операндами, яка створює новий об’єкт (результат).

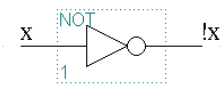
При виконанні булевих операцій змінні й результат набувають булеві значення 0 і 1.

Булеві функції можуть залежати від однієї, двох і в цілому n змінних. Булева функція від n змінних може бути задана не більше ніж на $N = 2^n$ наборах. Оскільки функції приймають тільки два значення, загальне число булевих функцій від n змінних обчислюється за формулою $M = 2^{2^n}$. Отже, функція від одного аргументу може мати чотири значення: $f_0(x) = 0$ (константа 0), $f_1(x) = x$ (еквівалентність), $f_2(x) = \bar{x}$ (інверсія x), $f_3(x) = 1$ (константа 1).

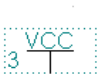
З наведених функцій в системі проектування MAX+plus II використовуються чотири f_0, f_1, f_2 та f_3, які реалізуються за допомогою примітивів (рис.2.1):

— 0 ,





— $\bar{x}$,



— 1.

Рисунок 2.1

Два аргументи надають можливість одержати 16 різних функції. Логічні функції від двох змінних наведено в таблиці 2.1.

Таблиця 2.1 – Структурні формули та назви логічних функцій

x_0	0	0	1	1	Функція	Назва функції
x_1	0	1	0	1		
I	0	0	0	0	$\mathbf{f_0 = 0}$	Константа 0
III	**0**	**0**	**0**	**1**	$\mathbf{f_1 = x_0 \wedge x_1}$	**Кон'юнкція (операція І)**
III	0	0	1	0	$\mathbf{f_2 = x_0 \wedge \bar{x}_1}$	Заборона по $\mathbf{x_1}$
II	0	0	1	1	$\mathbf{f_3 = x_0}$	Тотожно $\mathbf{x_0}$
III	0	1	0	0	$\mathbf{f_4 = \bar{x}_0 \wedge x_1}$	Заборона по $\mathbf{x_0}$
II	0	1	0	1	$\mathbf{f_5 = x_1}$	Тотожно $\mathbf{x_1}$
V	**0**	**1**	**1**	**0**	$\mathbf{f_6 = x_0 \oplus x_1 = (x_0 \wedge \bar{x}_1) \vee (\bar{x}_0 \wedge x_1)}$	**Сума за mod 2**
IV	**0**	**1**	**1**	**1**	$\mathbf{f_7 = x_0 \vee x_1}$	**Диз'юнкція (операція АБО)**
III	**1**	**0**	**0**	**0**	$\mathbf{f_8 = x_0 \downarrow x_1 = \overline{x_0 \vee x_1} = \bar{x}_0 \wedge \bar{x}_1}$	**Стрілка Пірса (оп-я АБО-НЕ)**
V	**1**	**0**	**0**	**1**	$\mathbf{f_9 = x_0 \sim x_1 = (x_0 \wedge x_1) \vee (\bar{x}_0 \wedge \bar{x}_1)}$	**Еквівалентність**
II	1	0	1	0	$\mathbf{f_{10} = \bar{x}_1}$	Інверсія $\mathbf{x_1}$
IV	1	0	1	1	$\mathbf{f_{11} = x_1 \rightarrow x_0 = x_0 \vee \bar{x}_1}$	Імплікація $\mathbf{x_1 \rightarrow x_0}$
II	1	1	0	0	$\mathbf{f_{12} = \bar{x}_0}$	Інверсія $\mathbf{x_0}$
IV	1	1	0	1	$\mathbf{f_{13} = x_0 \rightarrow x_1 = \bar{x}_0 \vee x_1}$	Імплікація $\mathbf{x_0 \rightarrow x_1}$
IV	**1**	**1**	**1**	**0**	$\mathbf{f_{14} = x_0 / x_1 = \overline{x_0 \wedge x_1} = \bar{x}_0 \vee \bar{x}_1}$	**Штрих Шиффера (оп-я І-НЕ)**
I	1	1	1	1	$\mathbf{f_{15} = 1}$	Константа 1

На рис. 2.2 наведено символи шести основних функцій: f_1 — AND2 (**І**), f_{14} — NAND2 (**І-НЕ**), f_7 — OR2 (**АБО**), f_8 — NOR2 (**АБО НЕ**), f_6 — XOR (**Виключаюче АБО**), f_9 — XNOR (**Виключаюче АБО-НЕ**), які використовуються в системі проектування MAX+plus II.

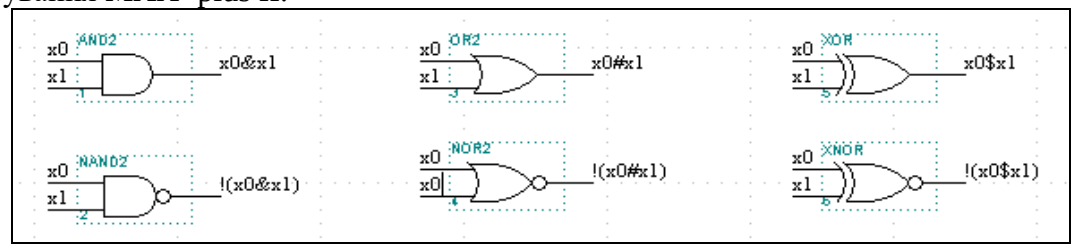


Рисунок 2.2 – Символи шести основних функцій

Наведені функції можна задати за допомогою таблиць істинності (табл. 2.2), у яких зліва подані значення операндів, а справа значення функції.

Таблиця 2.2 – Таблиці істинності основних функцій

Аргументи		Функції					
x_1	x_0	$f_1 = x_1 \,\&\, x_0$	$f_{14} = \overline{x_1 \,\&\, x_0}$	$f_7 = x_1 \# x_0$	$f_8 = \overline{x_1 \# x_0}$	$f_6 = x_1 \$ x_0$	$\overline{f_9 = x_1 \$ x_0}$
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

2. Приклад виконання лабораторної роботи

Дослідити логічну функцію $f_2 = x_0 \wedge \bar{x}_1$.

Графічну схему дослідження заданої функції, реалізовану в середовищі проектування MAX+plus II, наведено на рис. 2.3.

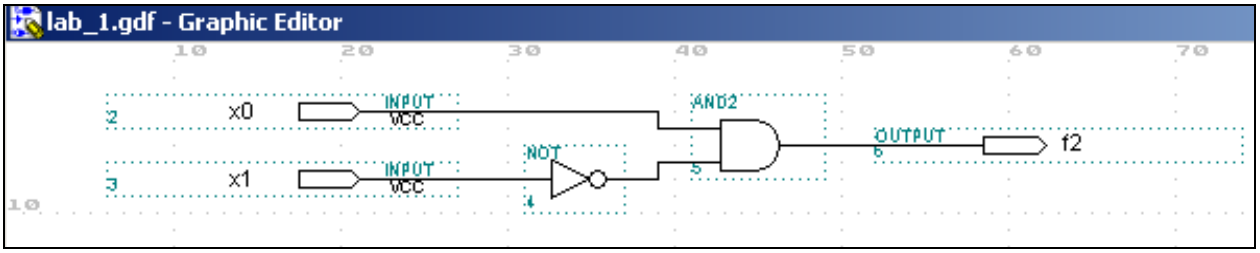


Рисунок 2.3 – Схема дослідження логічної функції

На рис. 2.4 наведено часові діаграми роботи схеми (результати моделювання).

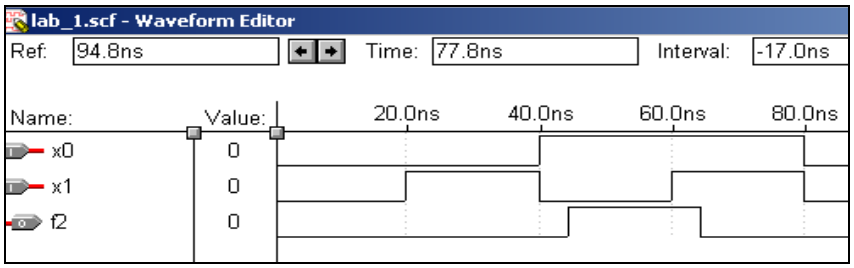


Рисунок 2.4 – Часові діаграм роботи схеми

Таблиця 2.3

x_0	x_1	$f_2 = x_0 \wedge \bar{x}_1$
0	0	0
0	1	0
1	0	1
1	1	0

Результати моделювання можна подати у вигляді таблиці істинності (табл. 2.3).

3. Порядок виконання роботи

3.1. Використовуючи систему проектування MAX+plus II, скласти схеми для дослідження логічних функцій, заданих згідно варіанту. Для побудови схем використовувати символ інвертора та символи функцій, наведених на рис. 2.1.

3.2. Для дослідження логічних функцій побудувати часові діаграми.

3.3. По одержаних часових діаграмах побудувати таблиці істинності.

Примітка. При дослідженні функцій f_6 і f_9 побудувати схему реалізації двома способами: за допомогою логічних функцій **І**, **АБО**, **НЕ** і **Виключаюче АБО** або **Виключаюче АБО-НЕ.**

4. Зміст звіту

4.1 Короткі теоретичні відомості.

4.2 Вихідні дані для виконання роботи.

4.3 Результати виконання роботи (схема, часові діаграми та таблиці істинності).

4.4 Висновки.

5. Індивідуальні завдання

Дослідити логічні функції двох змінних. Номер функції відповідає назві, згідно таблиці 2.4.

Таблиця 2.4 − Таблиця індивідуальних завдань

№ варі-анту	Номери функцій	№ варі-анту	Номери функцій	№ варі-анту	Номери функцій
1	1, 6, 8,13	11	6, 8, 10,14	21	2, 6, 11, 14
2	7, 9, 12, 14	12	1, 7, 9, 13	22	1, 5, 9, 13
3	6, 8, 11, 13	13	1, 6, 8, 10	23	2, 6, 10, 14
4	1, 9, 12, 14	14	8, 9, 13, 14	24	5, 9, 12, 14
5	1, 6, 8, 11	15	1, 5, 6, 11	25	2, 6, 11, 12
6	2, 7, 9, 12	16	3, 5, 7, 9	26	4, 5, 9, 13
7	5, 6 13, 14	17	2, 4, 6, 8	27	6, 8, 10, 11
8	1, 8, 9, 12	18	1, 3, 9, 11	28	3, 5, 9, 12
9	2, 4, 6, 14	19	6, 7, 10, 14	29	6, 10, 12, 14
10	8, 9, 10, 12	20	3, 8, 9, 10	30	4, 9, 11,13

Вказівка 1. Номер варіанту вибирається з табл. 2.4 за номером у журналі академічної групи.

Вказівка 2. При виконанні індивідуальних завдань в одному проекті можна досліджувати декілька функцій.

6. Контрольні запитання

6.1. Дайте визначення логічних функцій: інверсії, диз’юнкції, кон’юнкції.

6.2. Зобразіть умовні графічні позначення основних логічних елементів.

6.3. Запишіть таблиці істинності основних логічних елементів.

Лабораторна робота №3

Тема. Доведення основних законів алгебри логіки

Мета роботи: вивчення основних законів алгебри логіки та їх доведення з використанням системи проектування MAX+plus II.

Теоретичні відомості

1. Аксіоми та закони булевої алгебри

Булева алгебра базується на деяких аксіомах, із яких виводяться основні закони для перетворення виразів, що містять булеві змінні. Обґрунтування цих аксіом підтверджується таблицями істинності для розглядуваних операцій.

Кожна аксіома подана у двох формах: кон'юнктивній і диз'юнктивній, що випливає із принципу двоїстості логічних операцій, згідно якого операції кон'юнкції і диз'юнкції допускають взаємну заміну, якщо одночасно поміняти логічну 1 на 0, а 0 на 1; знак $\vee$ на $\wedge$, а знак $\wedge$ на $\vee$.

Аксіоми заперечення, кон'юнкції, диз'юнкції:

1. $\overline{0}=1;\ \overline{1}=0.$ 2. $0 \wedge 0 = 0;\ 0 \wedge 1 = 0;\ 1 \wedge 0 = 0;\ 1 \wedge 1 = 1.$

3. $0 \vee 0 = 0;\ 0 \vee 1 = 1;\ 1 \vee 0 = 1;\ 1 \vee 1 = 1.$

Основні закони булевої алгебри наведено в табл. 3.1.

Таблиця 3.1 – Закони булевої алгебри

№	Логічний вираз	Назва закону
1	$x_1 \wedge x_2 = x_2 \wedge x_1$	Комутативний закон
2	$x_1 \vee x_2 = x_2 \vee x_1$	
3	$x_1 \wedge (x_2 \wedge x_3) = (x_1 \wedge x_2) \wedge x_3$	Сполучний закон
4	$x_1 \vee (x_2 \vee x_3) = (x_1 \vee x_2) \vee x_3$	
5	$x_1 \wedge (x_2 \vee x_3) = (x_1 \wedge x_2) \vee (x_1 \wedge x_3)$	Розподільний закон
6	$x_1 \vee (x_2 \wedge x_3) = (x_1 \vee x_2) \wedge (x_1 \vee x_3)$	
7	$x_1 \wedge (x_1 \vee x_2) = x_1$	Закон поглинання
8	$x_1 \vee (x_1 \wedge x_2) = x_1$	
9	$(x_1 \vee x_2) \wedge (x_1 \vee \overline{x}_2) = x_1$	Закон склеювання
10	$(x_1 \wedge x_2) \vee (x_1 \wedge \overline{x}_2) = x_1$	
11	$\overline{x_1 \wedge x_2} = \overline{x}_1 \vee \overline{x}_2$	Закон де Моргана
12	$x_1 \wedge x_2 = \overline{\overline{x}_1 \vee \overline{x}_2}$	
13	$\overline{x_1 \vee x_2} = \overline{x}_1 \wedge \overline{x}_2$	Закон де Моргана
14	$x_1 \vee x_2 = \overline{\overline{x}_1 \wedge \overline{x}_2}$	
15	$x \wedge \overline{x} = 0,\ x \wedge 1 = x$	Закон нуля і одиниці
16	$x \vee \overline{x} = 1,\ x \vee 0 = x$	
17	$\overline{\overline{x}} = x$	Закон подвійного заперечення

2. Двоїстість логічних функцій

Означення 1. Логічна функція $f^*(x_1, x_2, \ldots, x_n)$ називається ***двоїстою*** до функції $f(x_1, x_2, \ldots, x_n)$, якщо $f^*(x_1, x_2, \ldots, x_n) = \overline{f(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)}$.

Означення 2. Функція, двоїста сама до себе, тобто $f^* = f$, називається ***самодвоїстою***.

Правило побудови двоїстої функції. Для запису функції f^*, двоїстої до функції f, треба у функції f всюди 0 замінити на 1, 1 – на 0, знак $\wedge$ – на $\vee$, а знак $\vee$ – на $\wedge$. Наведене правило побудови двоїстих функцій ***називається принципом двоїстості***.

3. Функції алгебри Жегалкіна

У деяких випадках, перетворення над формулами булевих функцій зручно виконувати користуючись алгеброю Жегалкіна.

В алгебрі Жегалкіна розглядають функції: $f_{15} = 1$, $x_1 \wedge x_2$ та $x_1 \oplus x_2$.

На основі аксіом і законів алгебри Жегалкіна можна вивести правила перетворення функцій заперечення, кон'юнкцію і диз'юнкцію через функцію додавання за модулем 2 і навпаки:

3.1. $\bar{x}_1 = x_1 \oplus 1$;

3.2. $x_1 \wedge x_2 = (x_1 \oplus x_2) \oplus (x_1 \vee x_2)$;

3.3. $x_1 \vee x_2 = x_1 \oplus x_2 \oplus (x_1 \wedge x_2)$;

3.4. $x_1 \oplus x_2 = (\bar{x}_1 \wedge x_2) \vee (x_1 \wedge \bar{x}_2) = (x_1 \vee x_2) \wedge (\bar{x}_1 \vee \bar{x}_2)$.

4. Функція Шеффера

Функція Шеффера (/) – функція, яка виражається співвідношенням $x_1 / x_2 = \overline{x_1 \wedge x_2}$.

Можна показати, що функції заперечення, кон'юнкція і диз'юнкція виражаються через функцію Шеффера за формулами:

4.1. $\bar{x} = \overline{x \wedge x}$,

4.2. $x_1 \wedge x_2 = \overline{x_1 / x_2} = (x_1 / x_2) / (x_1 / x_2) = \overline{\overline{x_1 \wedge x_2} \wedge \overline{x_1 \wedge x_2}}$,

4.3. $x_1 \vee x_2 = \overline{\bar{x}_1 \wedge \bar{x}_2} = \bar{x}_1 / \bar{x}_2 = x_1 / x_1 / x_2 / x_2 = \overline{\overline{x_1 \wedge x_1} \wedge \overline{x_2 \wedge x_2}}$.

Це означає, що функція Шеффера є універсальною функцією, яка записується як І-НЕ і через яку можна виразити будь-яку логічну функцію.

5. Функція Пірса

Функція Пірса (Вебба) (↓) – функція, яка виражається співвідношенням $x_1 \downarrow x_2 = \overline{x_1 \vee x_2} = \bar{x}_1 \wedge \bar{x}_2$.

Можна показати, що функції заперечення, кон'юнкція і диз'юнкція виражаються через функцію Пірса (Вебба) наступним чином:

5.1. $\bar{x} = x \downarrow x = \overline{x \vee x}$,

5.2. $x_1 \vee x_2 = (x_1 \downarrow x_2) \downarrow (x_1 \downarrow x_2) = \overline{\overline{x_1 \vee x_2} \vee \overline{x_1 \vee x_2}}$,

5.3. $x_1 \wedge x_2 = (x_1 \downarrow x_1) \downarrow (x_2 \downarrow x_2) = \overline{\overline{x_1 \vee x_1} \vee \overline{x_2 \vee x_2}}$.

Функція Пірса є універсальною функцією, яка записується як АБО-НЕ і через яку можна виразити будь-яку логічну функцію.

6.1. Порядок виконання індивідуальних завдань

Для доведення законів булевої алгебри необхідно побудувати схеми логічних рівнянь, правої та лівої частини закону і проаналізувати вихідні сигнали кожної схеми.

1. Використовуючи систему проектування MAX+plus II, скласти схеми лівої та правої частини закону (табл.3.2).
2. Побудувати часові діаграми дослідження законів алгебри логіки.
3. Проаналізувати вихідні сигнали двох схем.

6.2 Індивідуальні завдання

Дослідити закони булевої алгебри. Індивідуальні завдання наведено в табл. 3.2.

Таблиця 3.2 − Таблиця індивідуальних завдань

№ варі-анту	Номери законів	№ варі-анту	Номери законів	№ варі-анту	Номери законів
1	1, 7, 11, 15	11	5, 9, 13, 15	21	4, 10, 11, 15
2	2, 8, 12, 16	12	6, 10, 14, 16	22	3, 9, 12. 16
3	3, 9, 13, 15	13	1, 8, 11, 15	23	2, 7, 13. 15
4	4, 10,14, 16	14	2, 10, 12, 16	24	1, 8, 14, 16
5	5, 7, 11, 15	15	3, 9, 13, 15	25	6, 7, 11, 15
6	6, 8, 12, 16	16	4, 7, 14, 16	26	5, 9, 12, 16
7	1, 9, 13, 15	17	5, 7, 11, 15	27	4, 10, 13, 15
8	2, 10, 14, 16	18	6, 9, 12, 16	28	3, 8, 14, 16
9	3, 7, 11, 15	19	6, 8, 13, 15	29	2, 7, 11, 15
10	4, 8, 12, 16	20	5, 9, 14, 16	30	1, 9, 12, 16

Вказівка 1. Номер варіанту вибирається з табл.3.2 за номером у журналі академічної групи.

Вказівка 2. При виконанні індивідуальних завдань в одному проекті можна досліджувати декілька законів.

Завдання 6.3. Знайдіть двоїсті формули до таких функцій:

1) $(x \wedge (y \vee z)) \vee \bar{x} \wedge \bar{y})$;

2) $x \wedge \bar{y} \vee x \vee y \vee z \wedge t$.

Правильність формул перевірити за допомогою таблиць істинності або схеми, реалізованої з використанням системи проектування MAX+plus II.

Завдання 6.4. Визначте, чи є нижче наведені функції самодвоїстими:

1) $f(x, y) = (\bar{x} \wedge y) \wedge (x \wedge \bar{y})$;

2) $f(x, y, z) = (x \wedge y) \vee (x \wedge z) \vee (z \wedge y)$;

3) $f(x, y, z) = (\bar{x} \wedge \bar{y}) \vee (x \wedge z) \vee (\bar{z} \wedge \bar{y})$;

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

Правильність формул перевірити за допомогою таблиць істинності або схем, реалізованої з використанням системи проектування MAX+plus II.

Завдання 6.5. Користуючись системою проектування MAX+plus II дослідити правильність формул 3.1− 3.4, наведених в пункті 3.

Завдання 6.6. Користуючись системою проектування MAX+plus II дослідити правильність формул 4.1− 4.3, наведених в пункті 4.

Завдання 6.7. Користуючись системою проектування MAX+plus II дослідити правильність формул 5.1− 5.3, наведених в пункті 5.

7. Зміст звіту

7.1. Короткі теоретичні відомості з кожного пункту.
7.2. Вихідні дані для виконання роботи.
7.3. Результати виконання роботи (таблиці істинності, схеми та часові діаграми).

8. Контрольні запитання

8.1. Дайте визначення двоїстої та самодвоїстої функцій.
8.2. Наведіть основні закони булевої алгебри.
8.3. Які функції розглядають в алгебрі Жегалкіна?
8.4. Показати, що функції заперечення, кон’юнкція і диз’юнкція виражаються через функцію Шеффера та Пірса.

Лабораторна робота №4

Тема: Синтез комбінаційних схем

Мета роботи: навчитися розробляти функціональні схеми цифрових пристроїв з використанням системи проектування MAX+plus II.

Теоретичні відомості

1. Способи задання логічних функцій

Як відомо, існують різні способи задання логічних функцій. У попередніх роботах був розглянутий табличний спосіб, при якому кожному набору значень змінних в таблиці істинності вказується значення самої логічної функції. Цей спосіб наглядний і може бути використаний для запису функцій від будь-якої кількості змінних. Але при аналізі властивостей функцій алгебри логіки такий запис не є компактним. Тому для задання логічних функцій використовують аналітичний запис у вигляді формул .

Функціональну схему логічного пристрою одержують в результаті абстрактного синтезу, який складається з наступних етапів:

1) текстовий опис функцій логічного пристрою;

2) складання таблиці істинності за текстовим описом;

3) запис логічної функції логічного пристрою у вигляді досконалої нормальної диз'юнктивної форми (ДНДФ) або досконалої нормальної кон'юнктивної форми (ДНКФ);

4) мінімізація логічної функції;

5) вибір одного із логічних базисів для реалізації функціональної схеми;

6) перетворення логічного рівняння з використанням правил де Моргана;

7) побудова функціональної схеми цифрового пристрою.

8) моделювання роботи цифрового логічного пристрою.

2. Порядок виконання роботи

2.1. За текстовим описом функціонування логічного пристрою скласти таблицю істинності .

2.2. Записати логічну функцію у вигляді ДНДФ .

2.3. Розробити функціональну схему логічного пристрою .

2.4. Мінімізувати одержану логічну функцію у булевому базисі І, АБО, НЕ.

2.5. Перетворити одержану мінімальну форму з використанням правил Де-Моргана до заданого базису.

2.6. Розробити функціональні схеми за мінімізованими функціями у базисі І -НЕ , АБО-НЕ.

2.7. За допомогою хвильового редактора Waveform editor системи MAX+plus II, побудувати часові діаграми, які мають відображати всі можливі стани вхідних сигналів.

3. Приклад виконання лабораторної роботи

Синтезувати логічний пристрій з трьома вхідними змінними, який генерує сигнал "1" на виході, якщо не менше як дві підряд змінні приймають значення "1".

3.1 Складаємо таблицю істинності (табл.4.1).

Таблиця 4.1.

A	B	C	F(A,B,C)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

3.2. Логічне рівняння в вигляді ДНДФ представляє собою диз'юнкцію кон'юнкцій тих вхідних наборів, для яких *F*=1:

$$F(A,B,C) = \overline{A}BC + AB\overline{C} + ABC$$

3.3. Мінімізація логічної функції здійснюється шляхом використання законів алгебри логіки:

$$F(A,B,C) = BC(\overline{A} + A) + AB(\overline{C} + C) = BC + AB\,.$$

3.4. Функціональну схему реалізуємо в базисі І -НЕ, для цього мінімізоване рівняння перетворимо за законом Де Моргана:

в базисі І –НЕ: $$F(A,B,C) = \overline{\overline{BC + AB}} = \overline{\overline{BC} \cdot \overline{AB}}\,,$$

в базисі АБО –НЕ: $$F(A,B,C) = \overline{\overline{BC} + \overline{AB}} = \overline{\overline{\overline{B} + \overline{C}} + \overline{\overline{A} + \overline{B}}}\,.$$

3.5. Функціональні схеми та часові діаграми логічного пристрою, реалізованого у базисах І-НЕ, наведено на рис. 1, 2, а в базисі АБО-НЕ — на рис. 3, 4.

Зауважимо, що при реалізації функціональної схеми в базисі АБО-НЕ на виході додатково потрібно поставити інвертор.

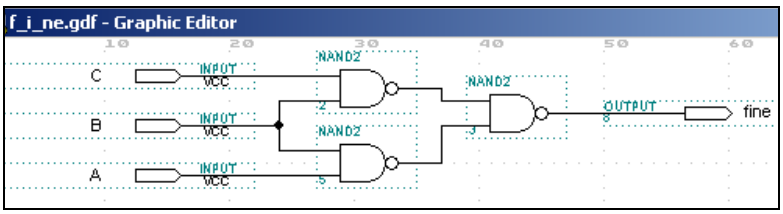


Рисунок 4.1 – Функціональна схема логічного пристрою у базисі І-НЕ.

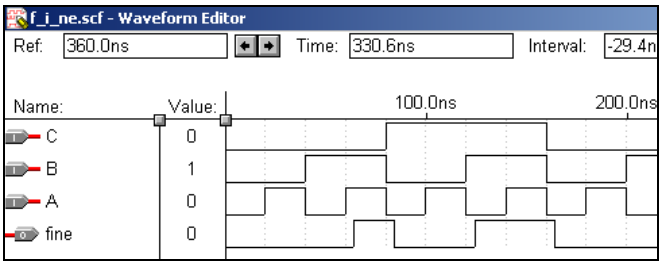


Рисунок 4.2 – Часові діаграми логічного пристрою у базисі І-НЕ

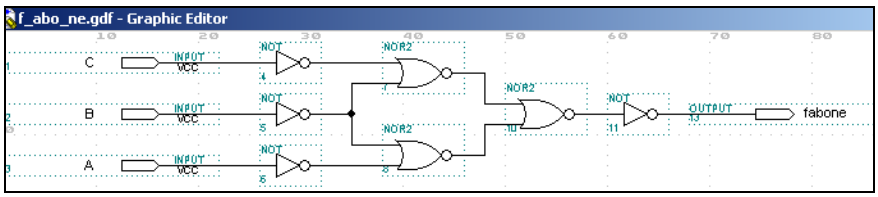


Рисунок 4.3 – Функціональна схема логічного пристрою в базисі АБО-НЕ

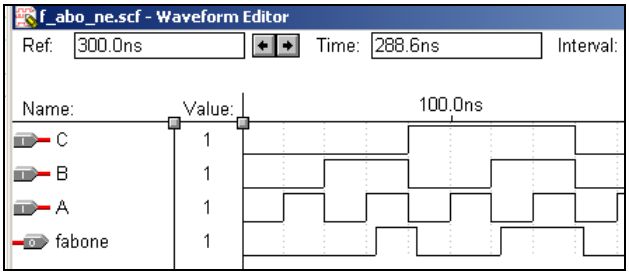


Рисунок 4.4 – Часові діаграми логічного пристрою в базисі АБО-НЕ

Висновки. З одержаних результатів можна зробити висновок, що функціональна схема логічного пристрою в базисі І-НЕ є простішою за схему пристрою в базисі АБО-НЕ.

4. Зміст звіту

4.1. Тема і мета роботи.
4.2. Вихідні дані для виконання роботи.
4.3. Результати виконання пунктів 2.1 – 2.7.
4.4. Функціональні схеми у базисах І-НЕ, АБО-НЕ.
4.5. Часові діаграми роботи пристроїв.
4.6. Висновки.

5. Індивідуальні завдання

Синтезувати цифровий пристрій з чотирма вхідними змінними, який генерує на виході сигнал “1”, тільки при вхідних значеннях змінних, що вказані у таблиці. На вхід комбінаційної схеми поступає 4-х розрядний паралельний двійковий код.

Таблиця 4.2 – Варіанти індивідуальних завдань

№ в-ту	F(A,B,C,D)=1	№ в-ту	F(A,B,C,D)=1	№ в-ту	F(A,B,C,D)=1
1	(0,4,12,14)	11	(2,5,10,15)	21	(1,4,12,14)
2	(1,5,11,14)	12	(1,4,11,15)	22	(2,5,10,14)
3	(2,6,10,13)	13	(0,3,7,12)	23	(3,6,10,13)
4	(3,7,9,12)	14	(1,2,13,15)	24	(0,7,9,12)
5	(4,8,11,13)	15	(1,3,9,14)	25	(1,8,11,13)
6	(5,7,9,10)	16	(0,4,12,14)	26	(2,7,9,10)
7	(2,6,9,10)	17	(0,5,13,15)	27	(2,6,9,15)
8	(5,7,8,11)	18	(2,3,13,14)	28	(5,7,8,14)
9	(4,7,8,12)	19	(4,5,12,15)	29	(0,7,8,12)
10	(3,6,9,13)	20	(3,5,14,15)	30	(1,6,9,13)

6. Контрольні запитання

6.1. Дайте визначення комбінаційного цифрового пристрою.
6.2. Назвіть етапи синтезу цифрових комбінаційних пристроїв.
6.3. Як записується досконала нормальна диз’юнктивна форма?
6.4. Як записується досконала нормальна кон’юнктивна форма?
6.5. Як здійснюється перетворення заданої функції в базис І-НЕ, АБО-НЕ?
6.6. Як здійснюється синтез функціональної схему пристрою за заданим функціями?

Лабораторна робота № 5

Тема: Мінімізація булевих функцій

Мета роботи: набути навички мінімізації функцій різними методами: алгебраїчним, Квайна, Квайна-Мак-Класкі, карт Карно та К-карт; вміти здійснювати моделювання з використанням системи проектування MAX+plus II.

Теоретичні відомості

1. Аналітичні методи мінімізації функцій

1.1 Основні поняття та означення

У загальному випадку ***мінімізацією*** логічної функції прийнято називати процес відшукання найменшого за складністю аналітичного зображення даної функції у вигляді суперпозиції функцій, які належать деякій функціонально повній системі.

Методи мінімізації розробляються стосовно кожної функціонально повної системи елементарних логічних функцій. Найдокладніше такі методи розроблено для випадку, коли функціонально повна система елементарних логічних функцій складається з диз'юнкції, кон'юнкції та заперечення. При цьому мінімізація логічної функції зводиться до визначення такої її форми, яка має найменший індекс (коефіцієнт) простоти, що характеризує складність ДНФ.

Індексом простоти $L(f)$ зображення функції $f(x_1, x_2, \ldots, x_n)$ у вигляді ДНФ (ДКФ) називається таке найменше число *L*, для якого існує формула, що зображає дану функцію і має точно *L* відповідних елементів, які покладено в основу мінімізації.

Найчастіше вживаються такі типи індексів простоти:

– $L_1(f)$ – сумарна кількість букв, (а не змінних), що до неї входять;

Логічна функція g називається ***імплікантою*** логічної функції f, якщо множина одиничних наборів функції g збігається або є підмножиною множини одиничних наборів функції f.

Простою імплікантою функції *f* називається будь-яка елементарна кон'юнкція *g*, яка є імплікантою цієї функції та має ту властивість, що ніяка її власна частина уже не є імплікантою даної функції.

Із визначення імпліканти випливає, що диз'юнкція будь-якого скінченого числа імплікант функції f є також імплікантою функції *f*.

Якщо при цьому імпліканти $g_1, g_2, \ldots, g_m$ сукупно накривають усі одиниці функції *f*, то має місце рівність $f = g_1 \vee g_2 \vee \ldots \vee g_m$. Таку систему імплікант прийнято називати ***повною***.

Диз'юнкцію всіх простих імплікант логічної функції називається ***скороченою диз'юнктивною нормальною формою*** СДНФ.

Знаходження скорочених ДНФ є першим етапом знаходження мінімальних форм булевих функцій. Як уже відмічалось, в скорочену ДНФ входять усі прості імпліканти булевої функції. Іноді із скороченої ДНФ можна вилучити одну або декілька простих імплікант, не порушуючи умови еквівалентності вихідній булеві функції. Такі прості імпліканти називаються ***лишніми***. Вилучення лишніх простих імплікант із скороченої ДНФ – другий етап мінімізації.

Диз'юнкцію простих імплікант, які складають зведену систему імплікант логічної функції, називається її ***тупиковою диз'юнктивною нормальною формою*** – ТДНФ.

Логічна функція може бути подана однією або декількома ТДНФ.

Мінімальною диз'юнктивною нормальною формою МДНФ функції f називається така ДНФ, яка зображає дану функцію і містить найменшу кількість букв порівняно з усі іншими еквівалентними їй ДНФ. Побудова тупикових і знаходження мінімальних ДНФ є третім етапом мінімізації.

Слід відмітити, що існують функції, які мають декілька мінімальних форм.

Таким чином, у загальному випадку, схема мінімізації логічних функцій складається з таких етапів: знаходження простих імплікант, побудови скороченої ДНФ функції, побудові тупикових ДНФ, з яких вибираються мінімальні.

Розглянемо ряд методів, які дають можливість виконати мінімізацію логічної функції за критерієм $L_1(f)$.

1.2 Алгебраїчний метод

Суть цього методу полягає в послідовному застосуванні основних аксіом і законів булевої алгебри. Найчастіше для спрощення формул використовуються закони: склеювання $F \cdot A + F \cdot \overline{A} = F$ та поглинання $F \cdot \tilde{A} + F = F$, де $\tilde{A}$ – змінна в прямій або інверсній формі.

Приклад 1. Мінімізувати функцію $f(A,B,C) = \underbrace{\overline{A}\overline{B}C}_{1} + \underbrace{\overline{A}B\overline{C}}_{2} + \underbrace{\overline{A}BC}_{3} + \underbrace{A\overline{B}C}_{4} + \underbrace{ABC}_{5}$,

де нижні індекси показують номери доданків, які являють собою мінтерми.

Розв'язання. Виконаємо операцію склеювання таких пар мінтермів: 1 і 3 за змінною *B*; 2 і 3 за змінною *C*; 4 і 5 за змінною *B*. В результаті виконання операцій склеювання отримаємо скорочену формулу $f(A,B,C) = \underbrace{\overline{A}C}_{1} + \underbrace{\overline{A}B}_{2} + \underbrace{AC}_{3}$, в якій можна провести склеювання доданків 2 і 3 за змінною *A*. При цьому отримаємо формулу $f(A,B,C) = A\overline{B} + C$, яка є мінімальною формою.

1.3 Метод Квайна

При мінімізації за методом Квайна в базисі $\{-,\vee,\wedge\}$ вважають, що функція задана в ДНФ і використовується поняття первинної або простої імпліканти.

Метод Квайна полягає в застосуванні до імплікант, які входять в ДНФ функції f закону поглинання $Fx + F\overline{x} = F$.

Використовуючи операцію поглинання вдається знизити ранг термів. Ця процедура продовжується до тих пір, поки не залишиться жодного терму, який допускає поглинання з яким-небудь іншим термом. Терм над яким було здійснено поглинання відмічаються. Не відмічені терми представляють собою первинні імпліканти.

Метод Квайна виконується в декілька етапів. Продемонструємо його на прикладі.

Приклад 2. Мінімізувати функцію $f(x,y,z) = \overline{x}\overline{y}z + \overline{x}y\overline{z} + \overline{x}yz + x\overline{y}z + xyz$.

Розв'язання. **Е т а п 1. Знаходження первинних імплікант.** Складаємо таблицю (табл. 1) і знаходяться імпліканти третього і другого рангу, тобто знижується ранг членів, які входять в ДДНФ.

Таблиця 5.1

	Початкові терми	$\overline{x}\overline{y}z$	$\overline{x}y\overline{z}$	$\overline{x}yz$	$x\overline{y}z$	xyz
+	$\overline{x}\overline{y}z$	1		$\overline{x}z$	$\overline{y}z$	
+	$\overline{x}y\overline{z}$		1	$\overline{x}y$		
+	$\overline{x}yz$	$\overline{x}z$	$\overline{x}y$	1		yz
+	$x\overline{y}z$	$\overline{y}z$			1	xz
+	xyz			yz	xz	1

У табл. 5.1 та нижче (табл. 5.2) символом “+” позначені терми (імпліканти), до яких застосовувався закон поглинання.

Після цього складається таблиця (табл. 5.2), яка включає всі терми, які не брали участь у поглинанні (таких немає), а також первинні імпліканти другого рангу.

В загальному, складання таблиць продовжується до тих пір, поки можна застосувати закон поглинання.

Таблиця 5.2

		$\bar{x}z$	$\bar{y}z$	$\bar{x}y$	xz	yz
+	$\bar{x}z$	1			z	
+	$\bar{y}z$		1			z
	$\bar{x}y$			1		
+	xz	z			1	
+	yz		z			1

Складаємо наступну таблицю, яка включає усі терми, які не підлягли поглинанню на попередніх кроках (таким термом в даному випадку є терм $\bar{x}\bar{y}$), а також імпліканти 1-го рангу.

У розглядуваному прикладі можна дійти до первинних імплікант першого рангу (табл. 5.3).

Таблиця 5.3

	$\bar{x}y$	z
$\bar{x}y$	1	
z		1

В результаті етапу 1 одержуємо дві первинні імпліканти: $\bar{x}y$ і z.

Е т а п 2. Розставлення міток. Складається таблиця (табл. 5.4), число рядків, якої дорівнює числу первинних імплікант, а число стовпців співпадає з числом мінтермів (конституент одиниць) ДДНФ функції *f*. Якщо в деякий мінтерм ДДНФ входить яка-небудь із первинних імплікант, то на перетині відповідного стовпця і рядка ставиться мітка (символ “*”).

Таблиця 5.4

Первинні імпліканти	1	2	3	4	5
	$\bar{x}\bar{y}z$	$\bar{x}y\bar{z}$	$\bar{x}yz$	$x\bar{y}z$	xyz
$\bar{x}y$		*	*		
z	*		*	*	*

Е т а п 3. Знаходження істотних імплікант. Якщо в деякому стовпці таблиці знаходиться лише одна мітка, то первинна імпліканта у відповідному рядку є істотною, оскільки без неї не буде можливості одержати усю множину заданих мінтермів. У даному випадку такою імплікантою є z і $\bar{x}y$. Стовпці, які відповідають істотним імплікантам, із таблиці викреслюються. У даному випадку такими імплікантами є z і $\bar{x}y$. Завдяки імплікації z в табл. 5.4 можна викреслити стовпці 1, 3, 4, 5, а завдяки імплікації $\bar{x}y$ – стовпець 2. Зауважимо, що в першу чергу розглядаємо імплікацію, яка більше покриває конституент одиниці. В результаті такої дії приходимо до результату, коли покриті усі конституент одиниці, а оскільки імпліканти z і $\bar{x}y$ є первинні, то можна записати мінімальну форму у вигляді $f_{\min}(x,y) = \bar{x}y + z$.

1.4 Метод Квайна – Мак-Класкі

Метод Квайна – Мак-Класкі представляє собою формалізацію етапу пошуку первинних імплікант за методом Квайна. Формалізація зводиться до наступного:

1. Усі мінтерми (конституенти одиниці) із ДДНФ булевої функції f записуються відповідними двійковими номерами (наборами).

2. На першому кроці усі номери розбиваються на групи по кількості одиниць у наборі.

3. Оскільки сусідні групи відрізняються між собою по кількості одиниць на одиницю, то склеювання здійснюється тільки між номерами сусідніх груп. Номери, які брали участь у склеюваннях якимось чином відмічаються, наприклад, закреслюються.

4. Кожний із наборів може брати участь у декількох склеюваннях, як і в методі Квайна. Невідмічені після склеювання номери являються первинними імплікантами.

5. На другому кроці результати склеювання, одержані на першому кроці, розбиваються на групи по розташуванню символу (наприклад, ”*”), яким відмічені змінні, які брали участь у склеюванні і проводиться склеювання у межах виділених груп. Знову набори, які брали участь у склеюванні відмічаються і підлягають перегрупуванні по розташуванні символів ”*” для подальшого склеювання, а набори, які не брали участь в склеювання відносяться до первинних імплікант і т.д.

6. Знаходження мінімальних ДНФ здійснюється за допомогою імплікантної таблиці, як і в методі Квайна.

Приклад 3. Методом Квайна – Мак-Класкі мінімізувати функцію від чотирьох змінних $f_4(0,4,7,8,9,11,12,13,15)$, яка на вказаних наборах дорівнює одиниці.

Розв'язання.

1. Запишемо номери наборів у вигляді двійкових наборів (мінтермів): 0000, 0100, 0111, 1000, 1001, 1011, 1100, 1101, 1111.

2. Утворимо групи по кількості одиниць в наборі (табл. 5.5).

Таблиця 5.5

Номер групи	Двійкові номери конституент одиниці
0	~~0000~~
1	~~0100~~, 1~~000~~
2	~~1001~~, 1~~100~~
3	~~0111~~, 1~~011~~, 1~~101~~
4	~~1111~~

Таблиця 5.6

Номери груп	Результати склеювання (двійкові номери імплікант)
0–1	0*00, *000
1–2	*100, 100*, 1*00
2–3	10*1, 1*01, 110*
3–4	*111, 1*11, 11*1

3. Склеюємо набори із сусідніх груп табл. 5.5. Результати склеювання наведено в табл.5.6. Закреслюємо набори в табл. 5.5, які брали участь в склеюванні.

4. Згруповуємо імпліканти по розташуванню зірочки в однакових позиціях (табл.5.7) і виконуємо склеювання в утворених групах. Результати склеювання зведено в табл. 5.8.

Таблиця 5.7

Номери груп	Набори груп
1	~~*000~~, *~~100~~, ~~*~~111
2	~~0*00~~, 1~~*00~~, 1*~~01~~, ~~1~~*11
3	~~10*1~~, 1~~1*1~~
4	100*, 110*

Таблиця 5.8

двійкові номери імплікант
**00, *111
00, 11
1**1
1*0*

Таблиця 5.9

Двійкові номери імплікант
**00
*111
1**1
1*0*

5. Згруповуємо імпліканти по розташуванню зірочки в однакових позиціях і заносимо їх в табл. 5.9. Оскільки подальші склеювання неможливі, то записуємо прості імпліканти: **00, *111, 1**1, 1*0*.

6. Будуємо імплікантну матрицю (табл. 5.10).

Таблиця 5.10

Двійкові набори простих імплікант	Двійкові набори конституент одиниці								
	0000	0100	0111	1000	1001	1011	1100	1101	1111
**00	+	+		+			+		
*111			+						+
1**1					+	+		+	+
1*0*				+	+		+	+	

За імплікантою матрицею визначаємо сукупність двійкових наборів простих імплікант, які відповідають мінімальній ДНФ. Це набори **00, *111, 1**1.

За допомогою двійкових наборів простих імплікант записуємо мінімальну ДНФ у буквеному вигляді

$$f_{\min}(a,b,c,d) = \bar{c}\bar{d} + bcd + ad\ .$$

Зауважимо, що розбиття конституент на групи дозволяє зменшити число попарних порівнянь при склеюванні. Крім цього оперування з двійковими наборами дає можливість здійснити реалізацію методу на ЕОМ.

2. Метод мінімізуючих карт

Даний метод базується на графічному способі подання логічних функцій. Як правило, його використовують у випадку ручної (без застосування ЕОМ) мінімізації логічних функцій при невеликій кількості змінних (не більше п’яти, шести).

Суть цього методу зводиться до графічного подання булевих функцій у вигляді добре відомих у літературі карт Вейча та Карно. Нами пропонується ще один вид карт – К–карти, які мають простішу структуру і, що головне, на їх основі досить легко реалізувати напівавтоматичний тренажер мінімізації логічних функцій на ЕОМ.

Будь-яка із перерахованих карта представляє собою прямокутну або квадратну таблицю, що містить 2^n клітинок, кожній із яких ставиться у відповідність одна із елементарних кон’юнкцій, або, що теж саме, один із двійкових наборів.

В основу мінімізації за допомогою мінімізуючи карт покладено поняття сусідніх клітинок.

Дві клітини називаються ***сусідніми***, якщо вони геометрично сусідні, тобто мають спільну сторону. При цьому клітини крайнього лівого і крайнього правого стовпців таблиці, а також нижнього та верхнього рядків таблиці, вважаються так само сусідніми. Сусіднім клітинам відповідають сусідні набори, тобто набори, які різняться тільки в одній позиції, що відповідає тому самому розряду, або тій самій змінній..

Оскільки карти Карно і К-карти будуються за простішими алгоритмами ніж Карти Вейча, то розглянемо мінімізацією лише за цими картами.

2.1 Мінімізація за допомогою карт Карно

Карта Карно — це прямокутна або квадратна таблиця, число клітин в якій дорівнює 2^n – по кількості мінтермів (конституент одиниці) для логічної функції від *n*-змінних. В карті Карно робиться розмітка рядків і стовпців змінними, що відповідає коду Грея. Кожній клітині приписується номер певного набору значень аргументів. На рис. 1 – 3 подано карти Карно для двох, трьох і чотирьох змінних, де показано приклади розмітки рядків і стовпців, десяткові й двійкові номери клітин та відповідні їм мінтерми.

Зауважимо, що десяткові і двійкові номери відповідних клітин в картах Карно змінюються в залежності від кількості змінних, що спричиняє певні незручності.

а)

$a \setminus b$	0	1
0	0	1
1	2	3

б)

$a \setminus b$	0	1
0	00	01
1	10	11

в)

$a \setminus b$	0	1
0	$\bar{a}\bar{b}$	$a\bar{b}$
0	$\bar{a}b$	ab

Рис. 5.1

а)

$a \setminus bc$	00	01	11	10
0	0	1	3	2
1	4	5	7	6

б)

$a \setminus bc$	00	01	11	10
0	000	001	011	010
1	100	101	111	110

в)

$a \setminus bc$	00	01	11	10
0	$\bar{a}\bar{b}\bar{c}$	$\bar{a}\bar{b}c$	$\bar{a}bc$	$\bar{a}b\bar{c}$
1	$a\bar{b}\bar{c}$	$a\bar{b}c$	abc	$ab\bar{c}$

г)

$a \setminus bc$	00	01	11	10
0		1	1	
1	1		1	1

Рис. 5.2

а)

$ab \setminus cd$	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

б)

$ab \setminus cd$	00	01	11	10
00	0000	0001	0011	0010
01	0100	0101	0111	0110
11	1100	1101	1111	1110
10	1000	1001	1011	1010

в)

$ab \setminus cd$	00	01	11	10
00	$\bar{a}\bar{b}\bar{c}\bar{d}$	$\bar{a}\bar{b}\bar{c}d$	$\bar{a}\bar{b}cd$	$\bar{a}\bar{b}c\bar{d}$
01	$\bar{a}b\bar{c}\bar{d}$	$\bar{a}b\bar{c}d$	$\bar{a}bcd$	$\bar{a}bc\bar{d}$
11	$ab\bar{c}\bar{d}$	$ab\bar{c}d$	$abcd$	$abc\bar{d}$
10	$a\bar{b}\bar{c}\bar{d}$	$a\bar{b}\bar{c}d$	$a\bar{b}cd$	$a\bar{b}c\bar{d}$

Рис. 5.3

2.2 Мінімізація за допомогою К-карт

К-карти — різновидність мінімізуючих карт, в основу побудови яких покладено перехід від карти для (*n*-1)-ї змінної до карти для *n* змінних за допомогою принципу дзеркального відображення. На рис. 4–8 подано К-карти для однієї, двох,..., п’яти змінних, одержаних за допомогою наступних правил.

Як і карта Карно, К-карта для однієї змінної складається з $2^1 = 2$ клітин (рис.4), для двох змінних (рис. 5) карта складається з 2^2=4 клітин, для трьох змінних (рис. 3.15) – із $2^3 = 8$ клітин і т. д. Карта для *n* змінних містить у два рази більше клітин, ніж карта для (*n*-1) змінної. При цьому подвоєння кількості клітин здійснюється у відповідністю з діаграмою:

1↓ 2→3 ↓ 4 →5 ↓ 6→7 ↓ 8→9 ↓ 10→11 ↓ 12→...,

де цифрами позначають кількість змінних, а стрілки – напрямок подвоєння карти.

Зокрема, для одержання карти для функції від двох змінних, карту для функції від однієї змінної подвоюємо вертикально вниз за допомогою дзеркального відображення та додавання до відображених значень числа $2^1 = 2$. Для одержання карти для функції від трьох змінних, карту для функції від двох змінних подвоюємо горизонтально вправо за допомогою дзеркального відображення та додавання до відображених значень числа $2^2 = 4$, при переході до карти для функції від чотирьох змінних, карту для функції від трьох змінних подвоюємо вертикально вниз і т. д.

На рис. 5.4 – 5.6 показано утворення К-карт для *n = 1, 2* і *3*, нумерація їх клітин у десятковій і двійковій системах числення, а також мінтерми, які відповідають номерам клітин.

$\bar{a}$	a
0	$0 + 2^0$

$\bar{a}$	a
0	1

$\bar{a}$	a
$\bar{a}$	a

Рис. 5.4

а)

	$\bar{a}$	a
$\bar{b}$	0	1
b	2	3

б)

	$\bar{a}$	a
$\bar{b}$	00	01
b	10	11

в)

	$\bar{a}$	a
$\bar{b}$	$\bar{a}\bar{b}$	$a\bar{b}$
b	$\bar{a}b$	ab

Рис. 5.5

	$\bar{a}$	a	a	$\bar{a}$
$\bar{b}$	0	1	$1+2^2$	$0+2^2$
b	2	3	$3+2^2$	$3+2^2$
	$\bar{c}$	$\bar{c}$	c	c

	$\bar{a}$	a	a	$\bar{a}$
$\bar{b}$	0	1	5	4
b	2	3	7	6
	$\bar{c}$	$\bar{c}$	c	c

	$\bar{a}$	a	a	$\bar{a}$
$\bar{b}$	000	001	101	100
b	010	011	111	110
	$\bar{c}$	$\bar{c}$	c	c

	$\bar{a}$	a	a	$\bar{a}$
$\bar{b}$	$\bar{c}\bar{b}\bar{a}$	$\bar{c}\bar{b}a$	$c\bar{b}a$	$c\bar{b}\bar{a}$
b	$\bar{c}b\bar{a}$	$\bar{c}ba$	cba	$cb\bar{a}$
	$\bar{c}$	$\bar{c}$	c	c

Рис. 5.6

На рис. 5.7 показано утворення К-карти для *n = 4,* нумерація її клітин у десятковій, шістнадцятковій та двійковій системах числення, а також мінтерми, які відповідають номерам клітин.

	$\bar{a}$	a	a	$\bar{a}$	
$\bar{b}$	0	1	5	4	$\bar{d}$
b	2	3	7	6	$\bar{d}$
b	$2+2^3$	$3+2^3$	$7+2^3$	$6+2^3$	d
$\bar{b}$	$0+2^3$	$1+2^3$	$5+2^3$	$4+2^3$	d
	$\bar{c}$	$\bar{c}$	c	c	

	$\bar{a}$	a	a	$\bar{a}$	
$\bar{b}$	0	1	5	4	$\bar{d}$
b	2	3	7	6	$\bar{d}$
b	10	11	15	14	d
$\bar{b}$	8	9	13	12	d
	$\bar{c}$	$\bar{c}$	c	c	

	$\bar{a}$	a	a	$\bar{a}$	
$\bar{b}$	0	1	5	4	$\bar{d}$
b	2	3	7	6	$\bar{d}$
b	A	B	F	E	d
$\bar{b}$	8	9	D	C	d
	$\bar{c}$	$\bar{c}$	c	c	

	$\bar{a}$	a	a	$\bar{a}$	
$\bar{b}$	0000	0001	0101	0100	$\bar{d}$
b	0010	0011	0111	0110	$\bar{d}$
b	1010	1011	1111	1110	d
$\bar{b}$	1000	1001	1101	1100	d
	$\bar{c}$	$\bar{c}$	c	c	

	$\bar{a}$	a	a	$\bar{a}$	
$\bar{b}$	$\bar{d}\bar{c}\bar{b}\bar{a}$	$\bar{d}\bar{c}\bar{b}a$	$\bar{d}\bar{c}b\bar{a}$	$\bar{d}\bar{c}b\bar{a}$	$\bar{d}$
b	$\bar{d}\bar{c}b\bar{a}$	$\bar{d}\bar{c}ba$	$\bar{d}cba$	$\bar{d}cb\bar{a}$	$\bar{d}$
b	$d\bar{c}b\bar{a}$	$d\bar{c}ba$	$dcba$	$dcb\bar{a}$	d
$\bar{b}$	$d\bar{c}\bar{b}\bar{a}$	$d\bar{c}\bar{b}a$	$dc\bar{b}a$	$fdc\bar{b}\bar{a}$	d
	$\bar{c}$	$\bar{c}$	c	c	

Рис. 5.7

Запис К-карт у двійковій або шістнадцятковій системах числення полегшує виконувати операцію склеювання вмісту сусідніх клітин.

Алгоритм мінімізації за допомогою карт зводиться до наступного:

1. Зображають карту для *n* змінних і роблять розмітку її рядків і стовпців. У клітини таблиці, які відповідають мінтермам (одиничним наборам) функції, що мінімізується, записують одиницю.

2. На побудованій карті для функції від *n* змінних виділяють максимальні прямокутні або квадратні області, які об'єднують вибрані одиничні значення функції. Кожна область повинна містити 2^k, $k = 0,1,2,\ldots$ клітинок. Вибрані області можуть перетинатися, тобто одна або декілька клітинок можуть належати різним областям. Чим менше прямокутників і чим більше клітин у прямокутниках, тим краще покриття. З декількох варіантів вибирають той, у якого менший коефіцієнт покриття $z = r/s$, де r – загальне число прямокутників, s – їхня сумарна площа в клітинках.

4. Формули, отримані в результаті мінімізації, містять r елементарних кон'юнкцій (за числом прямокутників в покритті). Кожна кон'юнкція містить тільки ті змінні, які не змінюють свого значення в наборах, що склеюються у відповідному прямокутнику. Число змінних у кон'юнкції називається її рангом. При склеюванні двох сусідніх клітинок одержують ранг кон'юнкції $n-1$, чотирьох – $n-2$, восьми – $n-3$ і т.д.

Приклад 5. Користуючись картами Карно мінімізувати функції:

$f_3(a, b, c) = \bigvee_1(1, 2, 6, 7)$; $f_4(a, b, c, d) = \bigvee_1(0,1,2,4,5,6,9,11)$.

Розв'язання. Відповідні карти Карно наведено на рис. 5.8 а), і рис. 5.8 б).

б)

$a \setminus bc$	00	01	11	10
0	0	1	0	1
1	0	0	1	1

в)

$ab \setminus cd$	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	0	0	0	0
10	0	1	1	0

Рис. 5.8

На кожній із карт виділимо прямокутні області максимальної площі і виконаємо склеювання відповідних мінтермів.

У випадку а) маємо дві прямокутні області по дві клітини, яким відповідають логічні суми по два мінтерми і один прямокутник, який складається з однієї клітини, якій відповідає один мінтерм:

$\bar{a}b\bar{c} + ab\bar{c} = b\bar{c}$, $abc + ab\bar{c} = ab$, $\bar{a}\bar{b}c$.

Виконавши операції склеювання та логічне сумування, одержаних імплікант, дістанемо $f_{\min} = b\bar{c} + ab + \bar{a}\bar{b}c$.

У випадку б) маємо один квадрат і два прямокутники, які розташовані на крайніх стовпчиках, що охоплює чотири крайні мінтерми і одна прямокутна область із двох клітин, якому відповідають логічні суми двох мінтермів:

$\bar{a}\bar{b}\bar{c}\bar{d} + \bar{a}\bar{b}\bar{c}d + \bar{a}b\bar{c}\bar{d} + \bar{a}b\bar{c}d = \bar{a}\bar{c}$, $\bar{a}\bar{b}\bar{c}\bar{d} + \bar{a}b\bar{c}\bar{d} + \bar{a}\bar{b}c\bar{d} + \bar{a}bc\bar{d} = \bar{a}\bar{d}$,

$a\bar{b}\bar{c}d + a\bar{b}cd = a\bar{b}d$.

Виконавши операції склеювання та логічне сумування, одержаних імплікант, дістанемо $f_{\min} = \bar{a}\bar{c} + \bar{a}\bar{d} + a\bar{b}d$.

Мінімізуємо задані логічні функції за допомогою К - карт.

За даними логічними функціями складемо К-карти (рис. 5.9 а) – 5.9 б)), в яких відображено двійкові набори, на яких функції приймають одиничні значення.

	$\bar{a}$	a	a	$\bar{a}$
$\bar{b}$		001		
b	010		111	110
	$\bar{c}$	$\bar{c}$	c	c

Рис.5.9 а)

	$\bar{a}$	a	a	$\bar{a}$	
$\bar{b}$	0000	0001	0101	0100	$\bar{d}$
b	0010			0110	$\bar{d}$
b		1011			d
$\bar{b}$		1001			d
	$\bar{c}$	$\bar{c}$	c	c	

Рис. 5.9 б)

Така форма подання функції є зручною для одержання склейок двійкових наборів (конституент одиниці), які входять в прямокутні області сусідні клітин.

Склеювання можна проводити у тих стовпчиках двійкових наборів, в яких кількість нулів і одиничок однакова. Склейки відмічено символом "*".

У табл. 5.11 наведено множини конституент одиниці, які входять в області

Таблиця 5.11

010 110	111 110	001
*10	11*	001
$b\bar{c}$	ab	$\bar{a}\bar{b}c$

Таблиця 5.12

0000 0001 0101 0100	0000 0010 0100 0110	 1011 1001
0*0*	0**0	1**0
$\bar{a}\,\bar{c}$	$\bar{a}\bar{d}$	$a\bar{b}d$

оптимального покриття одиничних значень функції $f_3(a,b,c)$ та відповідні склейки. Відповідна мінімальна форма має вигляд $f_{\min} = b\bar{c} + ab + \bar{a}\bar{b}c$.

Аналогічно в табл. 12 наведено множини конституент одиниці та відповідні склейки для функції $f_4(a,b,c,d)$. Відповідна мінімальна форма має вигляд $f_{\min} = \bar{a}\bar{c} + \bar{a}\bar{d} + a\bar{b}c$.

Наведемо також відповідні форми ДДНФ: $f_3(a,b,c) = \bar{a}\bar{b}c + \bar{a}b\bar{c} + ab\bar{c} + abc$,

$f_4(a,b,c,d) = \bar{a}\bar{b}\bar{c}\bar{d} + \bar{a}b\bar{c}d + \bar{a}\bar{b}c\bar{d} + \bar{a}bc\bar{d} + \bar{a}bcd + a\bar{b}\bar{c}d + a\bar{b}cd$.

Звернемо увагу: якщо записати функцію f_4 у вигляді ДДНФ, то для її реалізації потрібно було б 19 інверторів, 24 кон'юнктори та 7 диз'юнкторів. Після мінімізації потрібно лише 5 інверторів, 4 кон'юнктори і 2 диз'юнктори.

3. Завдання для індивідуальної роботи

Завдання 3.1. Задано функцію алгебри логіки від трьох змінних $f(x_1, x_2, x_3)$, яка на заданих десяткових наборах $f_3(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5)$ дорівнює одиниці. Для однієї з функції (згідно з варіантом, визначеним викладачем), наведеної в табл. 5.13, користуючись методами: алгебраїчним, Квайна, карт Карно та К-карт мінімізувати функцію. За допомогою системи проектування MAX+plus II побудувати відповідні схеми і провести моделювання їх роботи.

Варіанти індивідуальної роботи до завдання 3.1 наведено у табл. 5.13.

Таблиця 5.13

№ п/п	Функція $f_3(\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5)$		№ п/п	Функція $f_3(\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5)$
1	$f_3(0, 1, 2, 4, 6)$		**11**	$f_3(1, 3, 4, 5, 7)$
2	$f_3(0, 2, 3, 4, 6)$		**12**	$f_3(1, 3, 4, 5, 6)$
3	$f_3(0, 2, 4, 5, 6)$		**13**	$f_3(0, 1, 3, 5, 7)$
4	$f_3(0, 2, 4, 6, 7)$		**14**	$f_3(1, 2, 3, 5, 7)$
5	$f_3(0, 1, 2, 3, 5)$		**15**	$f_3(0, 4, 5, 6, 7)$
6	$f_3(0, 1, 2, 3, 4)$		**16**	$f_3(2, 4, 5, 6, 7)$
7	$f_3(0, 1, 2, 3, 7)$		**17**	$f_3(1, 4, 5, 6, 7)$
8	$f_3(0, 1, 2, 3, 6)$		**18**	$f_3(3, 4, 5, 6, 7)$
9	$f_3(1, 2, 4, 5, 7)$		**19**	$f_3(0, 2, 3, 5, 7)$
10	$f_3(0, 2, 3, 4, 6)$		**20**	$f_3(0, 1, 3, 4, 6)$

Завдання 3.2. Задано функцію алгебри логіки від чотирьох змінних $f(x_1, x_2, x_3, x_4)$, яка на заданих десяткових наборах $\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5,\alpha_6,\alpha_7,\alpha_8$ дорівнює одиниці, тобто $f_4(\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5,\alpha_6,\alpha_7,\alpha_8)=1$. Для однієї з функцій (згідно з варіантом, визначеним викладачем), наведеної в табл. 5.14, за допомогою методів Квайна-Мак-Класкі, карт Карно та К-карт мінімізувати функцію. За допомогою системи проектування MAX+plus II побудувати відповідні схеми і провести моделювання їх роботи.

Таблиця 5.14

№ п/п	Функція $f_4(\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5,\alpha_6,\alpha_7,\alpha_8)$		№ п/п	Функція $f_4(\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5,\alpha_6,\alpha_7,\alpha_8)$
1	$f_4(0, 1, 2, 3, 8, 10, 13, 15)$		**11**	$f_4(0, 1, 4, 5, 6, 7, 9, 11)$
2	$f_4(0, 2, 5, 7, 8, 9, 10, 11)$		**12**	$f_4(0, 1, 4, 5, 6, 7, 8, 10)$
3	$f_4(0, 1, 2, 3, 4, 6, 13, 15)$		**13**	$f_4(0, 1, 3, 4, 5, 7, 9, 11)$
4	$f_4(0, 2, 3, 8, 10, 11, 12, 14)$		**14**	$f_4(1, 5, 8, 9, 11, 12, 13, 15)$
5	$f_4(0, 1, 2, 3, 4, 5, 8, 10)$		**15**	$f_4(0, 4, 8, 9, 10, 12, 13, 14)$
6	$f_4(0, 1, 2, 3, 4, 5, 9, 11)$		**16**	$f_4(0, 4, 8, 9, 11, 12, 13, 15)$
7	$f_4(0, 1, 4, 5, 6, 7, 12, 14)$		**17**	$f_4(5, 7, 8, 9, 10, 11, 12, 13)$
8	$f_4(0, 1, 4, 5, 6, 7, 13, 15)$		**18**	$f_4(3, 4, 6, 7, 11, 12, 14, 15)$
9	$f_4(0, 1, 4, 5, 6, 10, 12, 14)$		**19**	$f_4(0, 2, 8, 9, 10, 11, 12, 13)$
10	$f_4(0, 1, 4, 5, 6, 8, 12, 15)$		**20**	$f_4(3, 4, 8, 9, 11, 12, 13, 15)$

Завдання 3.3. Задано функцію алгебри логіки від п'яти змінних $f(x_1, x_2, x_3, x_4, x_5)$, яка на заданих шістнадцяткових наборах дорівнює одиниці, тобто $f_5(\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5,\alpha_6,\alpha_7,\alpha_8,\alpha_9,\alpha_{10},\alpha_{11},\alpha_{12})=1$. Для однієї з функцій (згідно з варіантом, визначеним викладачем), наведеної в табл. 5.15, за допомогою методу К-карт мінімізувати функцію. За допомогою системи проектування MAX+plus II побудувати відповідні схеми і провести моделювання їх роботи.

Таблиця 5.15

№ п/п	Функція $f_5(\alpha_1,\alpha_2,\alpha_3,\ldots,\alpha_{11},\alpha_{12})$	№ п/п	Функція $f_5(\alpha_1,\alpha_2,\alpha_3,\ldots,\alpha_{11},\alpha_{12})$
1	00,01,04,05,04,14,15,11,10,09,0D,1D	**11**	0A,08,0B,09,1B,1A,19,18,00,01,11,10
2	02,03,07,06,16,17,13,12,0A,0B,1b,1A	**12**	05,04,07,06,14,15,16,17,0A,0B,1A,1B
3	0A,0B,0F,0E,1E,1F,1B,1A,02,03,12,13	**13**	)7,06,0F,0E,16,17,1E,1F,00,01,11,10
4	08,09,0D,0C,1C,1D,19,18,05,04,11,10	**14**	0F,0E,0D,0C,1E,1F,1C,1D,01,05,15,11
5	00,02,0A,08,10,12,1A,18,05,04,14,15	**15**	01,05,03,07,17,13,15,11,0E,)C,1E,1C
6	01,03,0B,09,11,13,1B,19,01,05,15,11	**16**	03,07,0B,0F,17,13,1F,1B,04,14,0C,1C
7	05,07,0F,0D,15,17,1F,1D,00,01,11,10	**17**	0F,0E,0D,0C,1E,1F,1C,1D,0!,03,11,13
8	04,06,0E,0C,14,16,1E,1C,07,06,16,17	**18**	05,04,14,15,07,06,16,17,09,0D,1D,19
9	00,02,01,03,11,10,12,13,0A,08,1A,18	**19**	06,16,0F,0E,16,17,!E,1F,02,0A,12,1A
10	02,03,0A,0B,13,12,1B,1A,06,16,0E,1E	**20**	0F,0E.0D,0C,1E,1F,1C,1D,01,03,11,13

Зауваження. У завданні 3.3 (табл. 5.15) кожний набір є шістнадцяткове число, перша цифра якого — двійкова цифра 0 або 1, а друга — шістнадцяткова цифра (відповідна двійкова тетрада).

4. Зміст звіту

4.1. Короткі теоретичні відомості про мінімізацію логічних функцій за допомогою методів: алгебраїчного, Квайна, Квайна -Мак-Класкі.

4.2. Короткі відомості про представлення та мінімізацію логічних функцій за допомогою карт Карно та К-карт.

4.3. Поетапні результати мінімізації заданих функцій за допомогою методу Квайна - Мак-Класкі, карт Карно та К-карт.

4.4. Функціональні цифрові логічні схеми, які відповідають як досконалим диз'юнктивним нормальним формам заданих функцій, так і результатам їх мінімізації.

Лабораторна робота № 6

Тема. Мінімізація неповністю визначених функцій

Мета роботи: Ознайомитись з методом мінімізації неповністю визначених функцій та оволодіти навичками подання логічних функцій у 8-ми базових формах з використанням логічних елементів І, АБО, НЕ; І-НЕ; АБО-НЕ. Роботу побудованих логічних функцій та відповідних схем перевірити за допомогою системи проектування MAX+plus II.

Теоретичні відомості

1. Неповністю визначені функції

Неповністю (або **частково**) **визначеними** логічними функціями називаються такі функції, які на частині двійкових наборів приймають значення нуль, на частині – одиниця, а на частині – значення функції невизначене. На останній частині наборів вибирається значення функції нуль або одиниця виходячи з умови, щоб нова функція була мінімальною серед усіх мінімальних функцій, які можна одержати при різних довизначеннях функції.

Завдання. Розробити логічні схеми для реалізації частково визначених логічних функцій F від 4-х аргументів, заданих таблицями істинності. Кожна комбінація значень аргументів двійкових змінних ABCD відображається числом N, яке дорівнює: $2^3A+2^2B+2^1C+ 2^0D$. Значення функцій при не вказаних комбінаціях значень аргументів необхідно довизначити для одержання схеми з мінімальним числом елементів. Мінімізацію логічної функції проводити за допомогою К-карт. Розробку схем провести на базі таких типів елементів:

- Елементи І, АБО, НЕ;
- Елементи І-НЕ;
- Елементи АБО-НЕ.

2. Приклад виконання лабораторної роботи

Мінімізувати неповністю визначену функцію, задану в табл. 6.1.

Таблиця 6.1

N	**4**	**6**	**7**	**8**	**9**	**11**	**12**	**13**	**14**	**15**
F	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**0**	**0**	**1**

Таблиця 6.2

N	**A**	**B**	**C**	**D**	**F**
4	**0**	**1**	**0**	**0**	**0**
6	**0**	**1**	**1**	**0**	**1**
7	**0**	**1**	**1**	**1**	**1**
8	**1**	**0**	**0**	**0**	**0**
9	**1**	**0**	**0**	**1**	**1**
11	**1**	**0**	**1**	**1**	**1**
12	**1**	**1**	**0**	**0**	**0**
13	**1**	**1**	**0**	**1**	**0**
14	**1**	**1**	**1**	**0**	**0**
15	**1**	**1**	**1**	**1**	**1**

Таблицю істинності заданої функції подано в табл.6.2. В табл.6.3 представлено К-карту вихідної функції, в якій символом “?” позначено клітини, що відповідають наборам, на яких функція невизначена. В табл. 6.4 представлено К-карту довизначеної функції, де заливкою позначені клітини, які відповідають наборам, на яких довизначена функція. Довизначення здійснюється так, щоб одержана функція була найбільш мінімальною із усіх можливих мінімальних форм довизначених функцій. Можливі склейки для довизначеної функції представлені в табл. 6.5. Мінімальна функція, одержана за допомогою К-карти має вигляд

$$F = \overline{B}D + \overline{A}C + CD\,.$$

Таблиця 6.3

?	?	?	0
?	?	7	6
?	11	15	0
0	9	0	0

Таблиця 6.4

0	0001	0	0
0010	0011	0111	0110
0	1011	1111	0
0	1001	0	0

Таблиця 6.5

0001	0010	0011
0011	0011	0111
1011	0111	1011
1001	0110	1111
*0*1	0*1*	**11

Нижче наведено перші чотири базові мінімальні форми знайденої мінімальної функції:

1) $F = \overline{A}C \vee \overline{B}D \vee CD$ — **форма І/АБО;**

2) $F = \overline{\overline{\overline{A}C} \wedge \overline{\overline{B}D} \wedge \overline{CD}}$ — **форма І-НЕ/І-НЕ;**

3) $F = \overline{(A \vee \overline{C}) \wedge (B \vee \overline{D}) \wedge (\overline{C} \vee \overline{D})}$ — **форма АБО/І-НЕ;**

4) $F = \overline{A \vee \overline{C}} \vee \overline{B \vee \overline{D}} \vee \overline{\overline{C} \vee \overline{D}}$ — **форма АБО-НЕ/АБО.**

На рис. 6.1- 6.4 наведено схеми, які відповідають знайденим формам 1) - 4):

1)

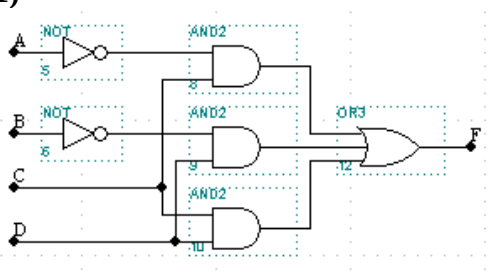


Рис. 6.1

2)

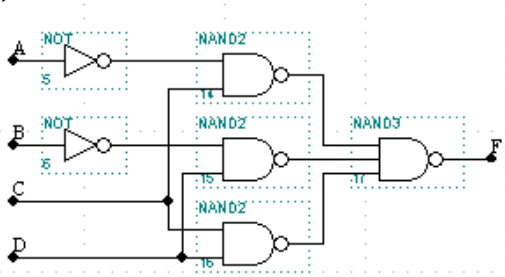


Рис. 6.2

3)

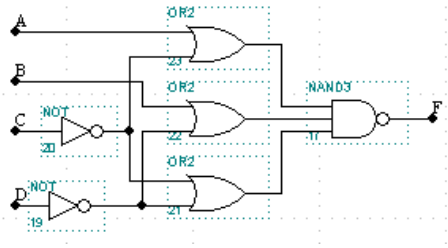


Рис. 6.3

4)

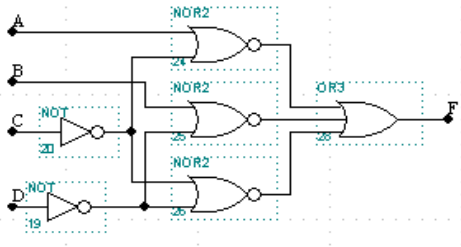


Рис. 6.4

В табл. 6.6 представлено К-карту функції $G = \overline{F}$. Можливі склейки для даної функції представлені в табл. 6.7. Мінімальна функція має вигляд

$$G = \overline{F} = \overline{C}\,\overline{D} + A\overline{D} + B\overline{C}\,.$$

Таблиця 6.6

0000		0101	0100
1010			1110
1000		1101	1100

Таблиця 6.7

0000	1010	0101
1000	1000	0100
0100	1110	1100
1100	1100	1101
**00	1**0	*10*

Нижче наведено наступні чотири базові мінімальні форми знайденої мінімальної функції:

5) $F = \overline{G} = \overline{\overline{C}\,\overline{D} + A\overline{D} + B\overline{C}}$ **— форма І/АБО-НЕ;**

6) $F = \overline{\overline{C}\,\overline{D}} \wedge \overline{A\overline{D}} \wedge \overline{B\overline{C}}$ **— форма І-НЕ/І;**

7) $F = (C \vee D) \wedge (\overline{A} \vee D) \wedge (\overline{B} \vee C)$ **— форма АБО/І;**

8) $F = \overline{\overline{C \vee D} \vee \overline{\overline{A} \vee D} \vee \overline{\overline{B} \vee C}}$ **— форма АБО-НЕ/АБО-НЕ.**

На рис. 6.5 – 6.8 наведено схеми, які відповідають знайденим мінімальним формам:

5)

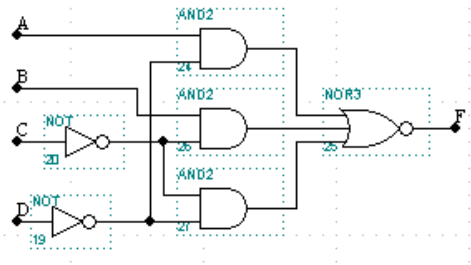


Рис. 6.5

6)

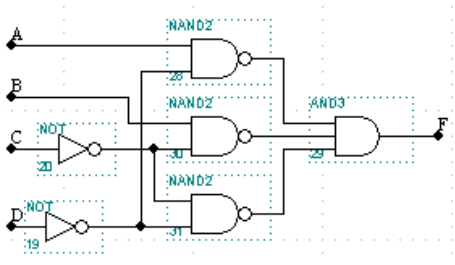


Рис. 6.6

7)

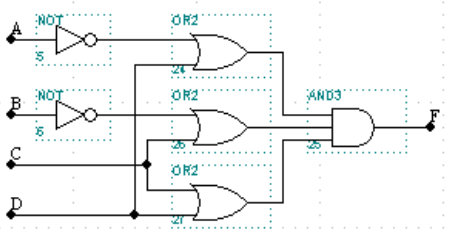


Рис. 6.7

8)

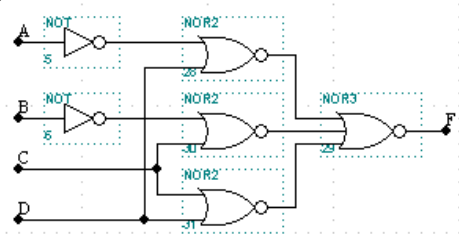


Рис. 6.8

Результати моделювання роботи схем 1) − 8) наведено на рис. 6.9.

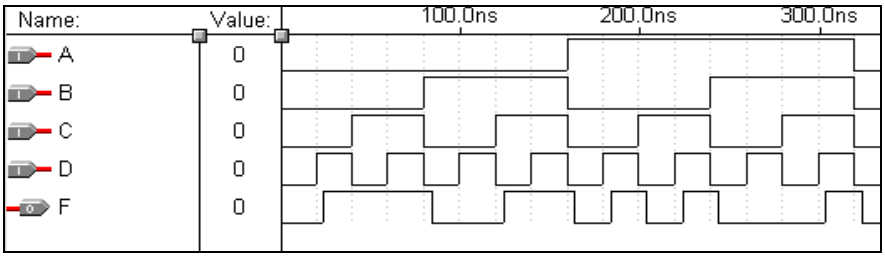


Рисунок 6.9 − Результати моделювання

2. Індивідуальні завдання

Варіанти індивідуальних завдань наведено в табл. 6.8.

Таблиця 6.8 − Варіанти індивідуальних завдань

1	N	1	2	3	4	6	7	8	9	11	12
	F	0	0	1	0	1	1	0	1	1	0
2	N	0	2	3	5	6	7	8	9	13	15
	F	0	1	0	0	1	1	0	0	1	0
3	N	1	2	3	4	6	7	9	12	13	14
	F	0	1	1	0	1	0	0	1	0	1
4	N	0	2	3	5	6	7	8	10	12	13
	F	0	1	1	0	0	1	1	1	0	0
5	N	0	1	3	4	6	9	10	11	14	15
	F	0	1	0	0	0	1	0	1	1	1
6	N	0	1	2	5	7	10	11	13	14	15
	F	0	0	0	1	0	1	0	1	1	1
7	N	1	3	4	5	6	10	11	12	14	15
	F	0	0	1	0	0	1	1	1	1	0
8	N	0	2	4	5	7	8	10	11	14	15
	F	1	0	0	0	0	1	1	1	0	1
9	N	0	1	3	4	5	6	9	10	11	14
	F	0	1	0	1	1	0	1.	0	1	0
10	N	0	1	2	4	5	7	10	11	13	15
	F	1	0	0	1	1	0	0	0	1	1
11	N	0	1	3	4	5	6	11	12	14	15
	F	1	1	0	1	0	0	0	1	1	0
12	N	0	1	2	4	5	7	8	10	14	15
	F	1	1	0	0	1	0	1	1	0	0
13	N	1	2	3	4	6	8	9	11	12	13
	F	0	0	1	0	0	0	1	1	1	1
14	N	0	2	3	5	7	8	9	12	13	15
	F	0	0	0	0	1	1	0	1	1	1
15	N	1	3	4	6	7	8	9	12	13	14
	F	0	0	0	1	0	1	1	1	0	1
16	N	0	2	5	6	7	8	9	10	12	13
	F	0	1	0	0	0	1	1	1	0	1
17	N	0	2	3	5	6	7	8	9	10	13
	F	0	1	0	0	1	1	1	0	1	0
18	N	1	2	3	4	6	7	8	9	12	14
	F	0	0	1	0	1	1	0	0	1	1
19	N	0	2	3	5	6	7	8	12	13	15
	F	0	1	1	0	0	1	0	0	1	1
20	N	1	2	3	4	6	7	9	11	12	13
	F	0	1	1	0	1	0	1	1	0	0

Лабораторна робота № 7

Тема: Комбінаційні схеми середнього ступеня інтеграції. Побудова напівсуматорів та суматорів.

Мета роботи: Набуття практичних навичок проектування напівсуматорів та суматорів з використанням пакету MAX+PLUS II.

Теоретичні відомості

1. Комбінаційні схеми середнього ступеня інтеграці

Комбінаційною схемою називається логічна схема, яка реалізовує однозначну відповідність між значеннями вхідних і вихідних сигналів. Для реалізації комбінаційних схем використовуються логічні елементи, які випускаються у вигляді інтегральних схем (ІС) дешифраторів, шифраторів, мультиплексорів, демультиплексорів, суматорів та інше.

1.1. Арифметичні суматори

Арифметичним суматором називається комбінаційна схема, яка призначена для додавання двох однорозрядних двійкових чисел.

Арифметичні суматори являються складовими частинами так званих арифметико-логічних пристроїв (АЛП) мікропроцесорів (МП), де вони використовуються як для підсумування двійкових чисел, так і для формування фізичної адреси комірки пам'яті.

На практиці використовуються суматори двох типів: напівсуматори і повні суматори.

1.2. Двійковий напівсуматор

Двійковий напівсуматор виконує додавання двох однорозрядних двійкових чисел. Він має два входи для доданків: A і B і два виходи: суми (S) і переносу (Co – Carry Out – перенос). Таблицю функціонування наведено в табл. 7.1.

Сумування здійснюється за допомогою логічної операції додавання за модулем два (S=A⊕B), а перенос за допомогою логічної операції кон'юнкції (Co=A∧B). Логічну схему суматора та його символ наведено на рис. 7.1.

Таблиця 7.1

A	B	S	Co	Примітка
0	0	0	0	0+0=0
0	1	1	0	0+1=1
1	0	1	0	1+0=1
1	1	0	1	1+1=0 (Carry=1)

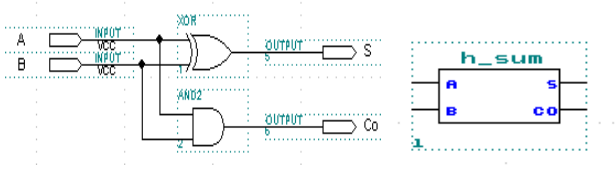


Рис. 7.1

1.3. Повний двійковий суматор (1-ий варіант)

Повний двійковий суматор виконує додавання трьох однорозрядних двійкових чисел. Він має три входи: для двох доданків A, B і розряду переносу C_i (Carry In) і два виходи: суми (S) і переносу (Co – Carry Out). Таблицю функціонування наведено в табл. 7.2.

Таблиця 7.2

A	B	C_i	Co	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Сумування та перенос здійснюються за допомогою логічних виразів:

$$S = \left(\overline{A} \wedge B \vee A \wedge \overline{B}\right) \wedge \overline{C_i} \vee \left(\overline{A} \wedge \overline{B} \vee A \wedge B\right) \wedge C_i;$$

$$S = (A \oplus B) \wedge \overline{C_i} \vee \overline{A \oplus B} \wedge C_i = A \oplus B \oplus C_i;$$

$$Co = \overline{A} \wedge B \wedge C_i \vee A \wedge \overline{B} \wedge C_i \vee A \wedge B \wedge C_i \vee A \wedge B \wedge \overline{C_i} = (A \oplus B) \wedge C_i \vee A \wedge B.$$

На рис. 7.2 наведено схему повного суматора, реалізованого на двох напівсуматорах і одному елементі “АБО”. Результати моделювання роботи однорозрядного повного суматора наведено на рис. 7.3. На рис. 7.4 наведено символ однорозрядного повного суматора.

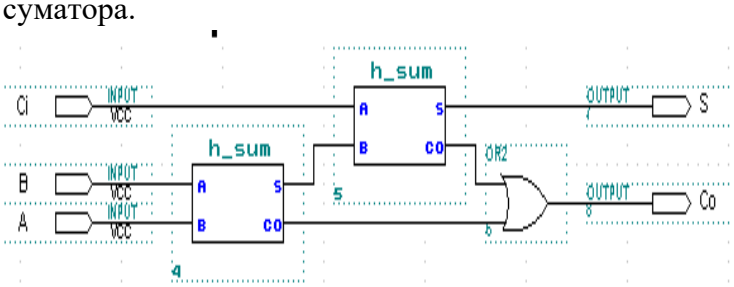


Рис. 7.2

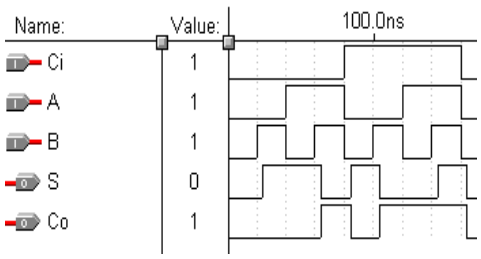


Рис. 7.3

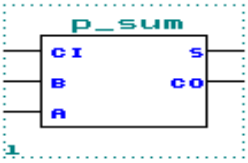


Рис. 7.4

Як приклад, на рис. 7.5 наведено схему діючої моделі чотирирозрядного двійкового суматора, реалізованого за допомогою пакету MAX+plus II.

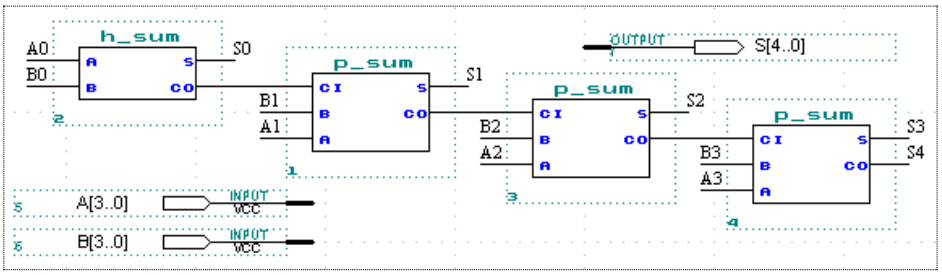


Рис. 7.5

На рис. 7.6 наведено результати моделювання роботи чотирирозрядного двійкового суматора.

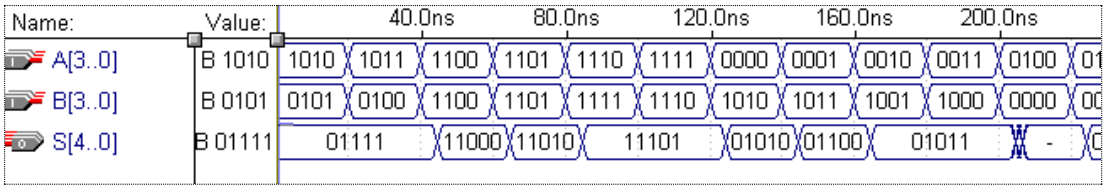


Рис. 7.6

1.4. Повний двійковий суматор (2-ий варіант)

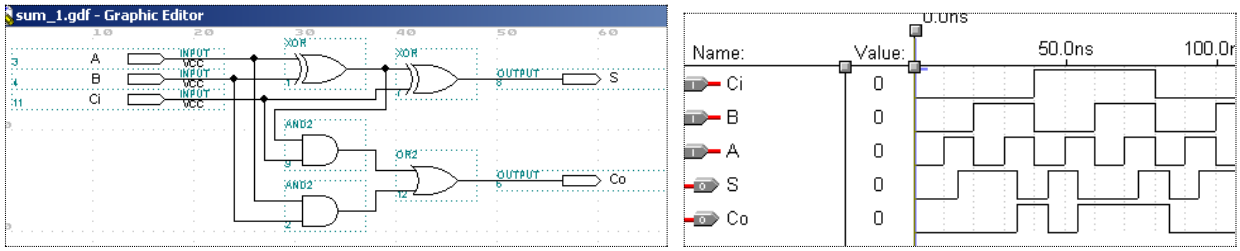


Рис. 7.7

1.5. Чотирирозрядний двійковий суматор

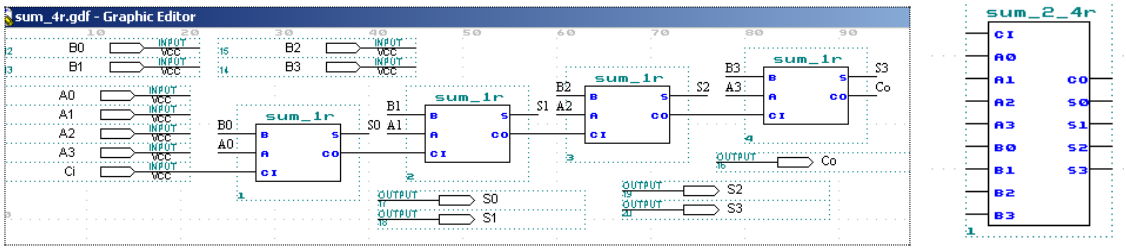


Рис. 7.8

На рис. 7.9 наведено результати моделювання роботи чотирирозрядного двійкового суматора.

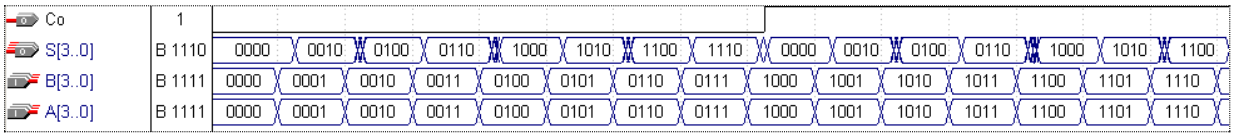


Рис. 7.9

2. Завдання для самостійної роботи

2.1. Користуючись символьним редактором пакету **MAX+plus II** побудувати логічну схему суматора та його символ, наведено на рис. 7.1.

2.2. Користуючись символьним редактором пакетом **MAX+plus II** побудувати схему повного суматора, реалізованого на двох напівсуматорах і одному елементі “АБО”, наведеного на рис.7.2.

2.3. Побудувати схему діючої моделі чотирирозрядного двійкового суматора, реалізованого за допомогою пакету MAX+plus II, наведену на рис. 7.5.

3. Зміст звіту

3.1. Короткі теоретичні відомості про арифметичні суматори, двійковий напівсуматор, повний двійковий суматор

3.2. Функціональні цифрові логічні схеми суматора, двійкового напівсуматора, повного двійкового суматора, чотирирозрядного двійкового суматора, результати моделювання в системі проектування MAX+plus II.

Лабораторна робота № 8

Тема: Типові вузли цифрової схемотехніки. Побудова шифраторів та дешифраторів.

Мета роботи: Набуття практичних навичок проектування шифраторів, дешифраторів з використанням пакету MAX+PLUS II .

Теоретичні відомості

1. Типові вузли цифрової схемотехніки

До типових вузлів належать логічні схеми, які найчастіше використовуються в цифрових ЕОМ. Серед них існують як комбінаційні так і послідовнісні схеми з пам'яттю.

До типових комбінаційних вузлів цифрової схемотехніки відносяться шифратори, дешифратори, мультиплексори і демультиплексори.

1.1. Дешифратори

Дешифратор (декодер) – це типова логічна комбінаційна схема з n інформаційними входами і 2^n виходами. Тобто це схема призначена для реалізації конституент одиниці.

Розрізняють *повні* і *неповні* дешифратори. Повні дешифратори реалізують 2^n конституент, де n – це число інформаційних входів. Неповні дешифратори реалізують менше ніж 2^n конституент.

Функціонування повного дешифратора описується системою логічних виразів вигляду:

$$F_0 = \overline{X}_{n-1}\overline{X}_{n-2}\ldots\overline{X}_1\overline{X}_0;$$
$$F_1 = \overline{X}_{n-1}\overline{X}_{n-2}\ldots\overline{X}_1 X_0;$$
$$\ldots$$
$$F_{2^n-1} = X_{n-1}X_{n-2}\ldots X_1 X_0,$$

де $X_{n-1}X_{n-2}\ldots X_0$ – вхідні двійкові змінні; F_0, F_1, ..., F_{2^n-1} – вихідні логічні функції, що являють собою мінтерми. Якщо дешифратор неповний, то число виходів m менше ніж 2^n.

Таблиця істинності функцій повного дешифратора, для $n=3$ наведена в табл.8.1:

Таблиця 8.1

X_2	X_1	X_0	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Індекс функції F_i визначає номер обраного виходу і відповідає десятковому еквіваленту вхідного коду. Вихід, на якому з'являється керуючий сигнал, називають ***активним***. Якщо значення сигналу на активному виході відображається логічною 1 (H), то на решті пасивних виходів встановлюється логічний 0 (L). Двійковий код, який завжди містить тільки одну одиницю, решта – нулі, називається ***унітарним***. Тому дешифратори є перетворювачами вхідного позиційного коду в унітарний вихідний.

У дешифраторах в інтегральному виконанні стан активного виходу часто відображається значення логічний 0 (L), а на інших пасивних виходах установлюється логічна 1 (H).

Функціонування повного дешифратора з інверсними виходами представляється логічними формулами вигляду:

$$G_0 = X_{n-1} \vee X_{n-2} \vee \ldots \vee X_1 \vee X_0;$$

$$G_1 = X_{n-1} \vee X_{n-2} \vee \ldots \vee X_1 \vee \overline{X}_0;$$

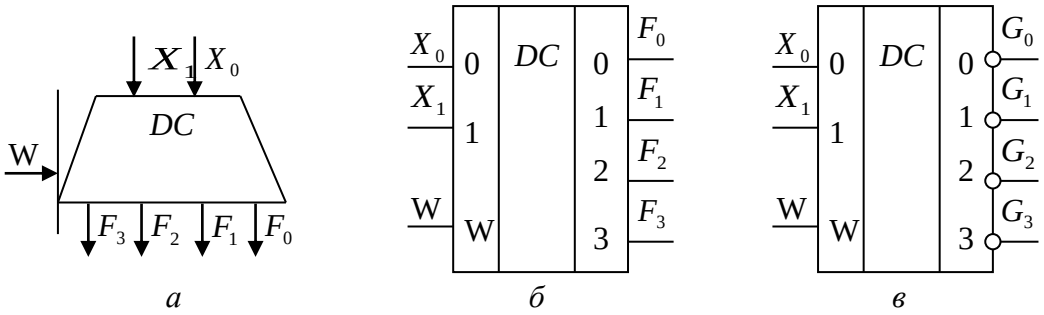
...

$$G_{2^n-1} = \overline{X}_{n-1} \vee \overline{X}_{n-2} \vee \ldots \vee \overline{X}_1 \vee \overline{X}_0,$$

де G_0, G_1, ..., G_{2^n-1} – вихідні логічні функції, що являють собою макстерми.

Умовні графічні позначення дешифраторів на електричних схемах показані на рис.8.1.

Логічна функція дешифратора позначається буквами DC (decoder). Мітки лівого додаткового поля в умовному позначенні відображають десяткові ваги вхідних комбінацій двійкових змінних, а мітки правого додаткового поля відповідають десятковим еквівалентам вихідних комбінацій двійкових змінних. У схему дешифраторів вбудовується один або два стробуючі (дозволяючі) входи, наприклад, *W* (рис. 1 б). За допомогою цього сигналу на вході визначають момент спрацювання дешифратора; крім того, вхід *W* використовується для нарощування розрядності вхідного коду.



а *б* *в*

Рисунок 8.1 — Умовні графічні позначення дешифратора:
а — на функціональних схемах; б, в — на принципових схемах

1.2. Побудова дешифраторів з використанням пакету MAX+plus II

Розглянемо методи побудови дешифраторів з використанням пакету MAX+plus II.

Приклад 1. Побудувати повний дешифратор при $n = 3$ користуючись пакетом MAX+plus II та дослідити його роботу.

На рис. 8.2 наведено схему повного дешифратора та результати моделювання його роботи.

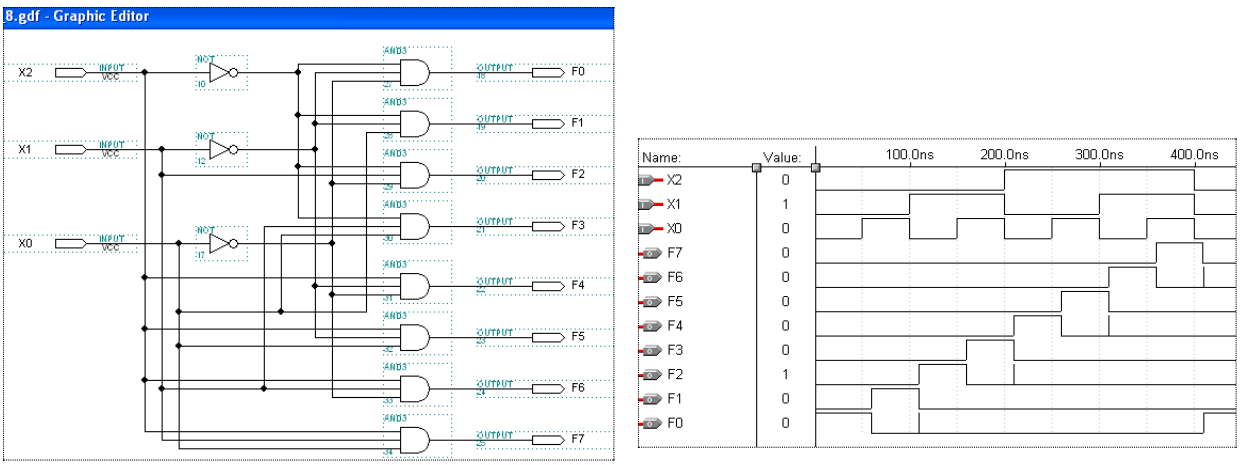


Рисунок 8.2

Приклад 2. Користуючись пакетом MAX+plus II побудувати дешифратор на два входи і чотири виходи, на одному з яких формується низький потенціал (логічний 0), на інших – високий (логічна 1). Нижче наведено таблицю істинності такого дешифратора (табл.8.2).

Таблиця 8.2

X_1	X_0	F_0	F_1	F_2	F_3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

Функції виходу для такого дешифратора, реалізовані в базисі І-НЕ, мають вигляд:

$$Y0 = \overline{\overline{X}_1 \cdot \overline{X}_0}\,,\ Y1 = \overline{\overline{X}_1 \cdot X_0}\,,\ Y2 = \overline{X_1 \cdot \overline{X}_0}\,,\ Y3 = \overline{X_1 \cdot X_0}\,.$$

Схема та результати моделювання такого дешифратора наведено на рис. 8.3.

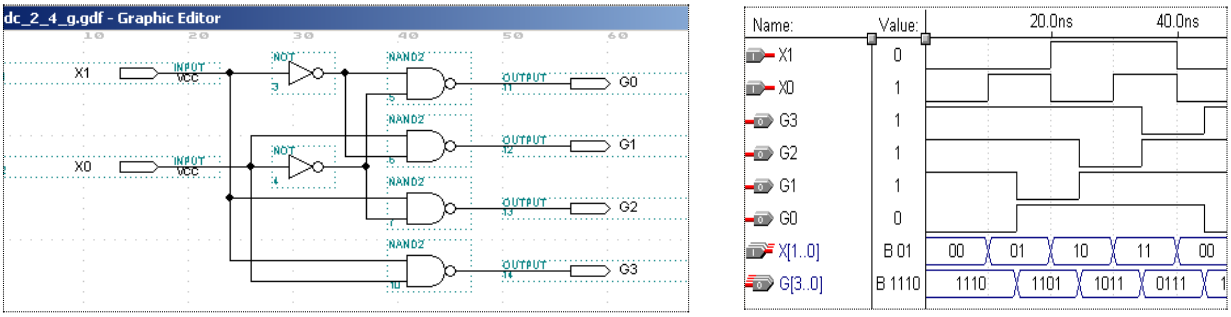


Рисунок 8.3 – Схема та результати моделювання дешифратора

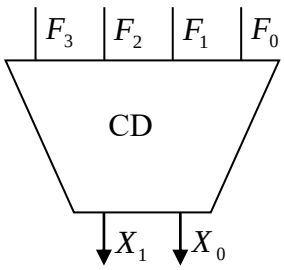
2. Шифратори

Шифратор – це типова комбінаційна схема, призначена для перетворення просторового унітарного коду (коду 1 із m) в двійковий код з природним порядком ваг.

Унітарний код, як уже відмічалось, має в своєму записі одну одиницю. З урахуванням цього можна вважати, що шифратор виконує функцію зворотну функції дешифратора, хоча в загальному випадку вихідний код може відрізнятися від коду з природним порядком ваг.

Повний двійковий шифратор має $m = 2^n$ входів і n виходів. Умовні графічні зображення шифратора при $n = 2$ на схемах наведені на рис. 8.4.

а)



б)

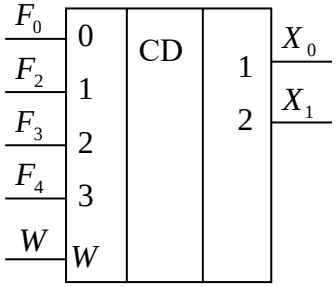


Рис. 8.4. Умовні графічні позначення шифратора:
а — на функціональних схемах; б, в — на принципових схемах.

Таблиці істинності шифраторів для дворозрядного і трирозрядного вихідних кодів з природним порядком ваг наведено у таблицях (табл. 8.3):

Таблиця 8.3 − Таблиці істинності шифраторів при $n=2$ і $n=3$

F_0	F_1	F_2	F_3	X_1	X_0
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	0

F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	X_2	X_1	X_0
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

На основі таблиць істинності можна записати логічні рівняння шифраторів:

$X_1 = F_2 \vee F_3$, $X_0 = F_1 \vee F_3$;

$X_2 = F_4 \vee F_5 \vee F_6 \vee F_7$, $X_1 = F_2 \vee F_3 \vee F_6 \vee F_7$, $X_0 = F_1 \vee F_3 \vee F_5 \vee F_7$.

2.1. Побудова шифраторів з використанням пакету MAX+plus II

Приклад 3. Побудувати шифратор, заданий вище наведеною таблицею істинності при $n=3$ та дослідити його роботу.

Функціональну схему шифратора, реалізовану на чотиривходових елементах АБО та його умовне графічне зображення, одержане шляхом редагування автоматично побудованого символу, наведено на рис.8.5. Результати моделювання роботи шифратора наведено на рис. 8.6. Звернемо увагу на те, що, оскільки, нумерація вхідних вузлів ведеться із 1 до 7, то вхід F_0 не використовується.

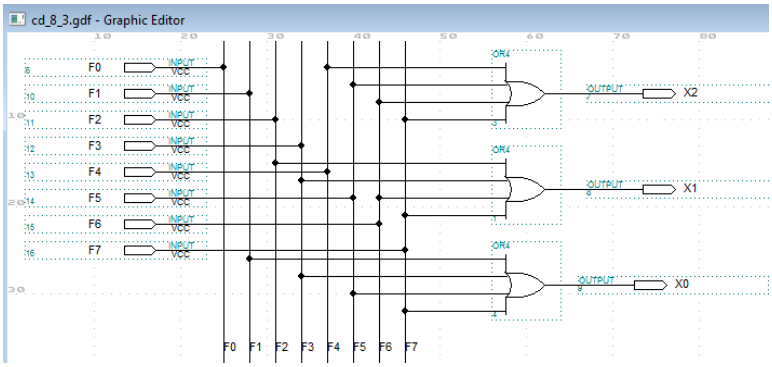


Рисунок 8.5 − Функціональна схема шифратора та його графічне зображення

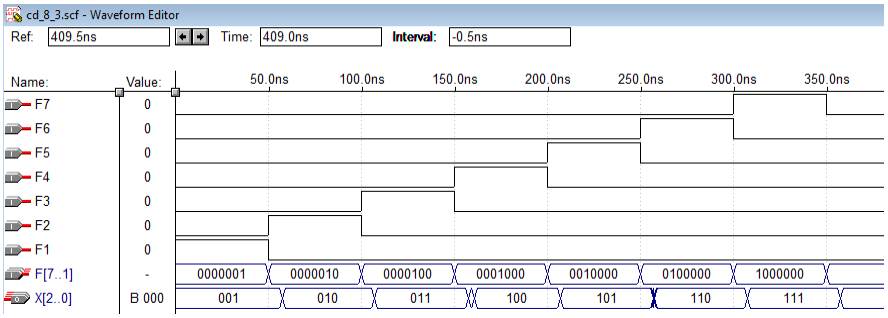


Рисунок 8.6 − Результати моделювання роботи шифратора

Одне із основних застосувань пріоритетного шифратора – введення даних з клавіатури, наприклад десяткових цифр. Натискання клавіші з десятковою цифрою 0 , 1, ..., 9 мають приводити до передачі в цифровий пристрій двійково-десяткового коду цієї цифри. Для цього використовують неповний шифратор “з 10 в 4 ”.

Шифратори, які при одночасному натискуванні декількох клавіш виробляють код тільки старшої цифри, називаються ***пріоритетними***. Логіка роботи пріоритетного шифратора на десять входів наведена в табл. 3, де $D_0, D_1, \ldots, D_9$ вхідні сигнали з десяткової клавіатури (наприклад, мікрокалькулятора); $A_3 A_2 A_1 A_0$ — вихідні сигнали, які задають двійково-десятковий код (двійкову тетроду) десяткової цифри.

Таблиця 8.4

D_i	A_3	A_2	A_1	A_0
D_0	0	0	0	0
D_1	0	0	0	1
D_2	0	0	1	0
D_3	0	0	1	1
D_4	0	1	0	0
D_5	0	1	0	1
D_6	0	1	1	0
D_7	0	1	1	1
D_8	1	0	0	0
D_9	1	0	0	1

Оскільки є можливість натиснення зразу декількох клавіш, то це може привести до збудження декількох вхідних шин, що в свою чергу приведе до видачі коду, який у загальному випадку, не буде співпадати з кодом жодної із натиснутих клавіш. Наприклад, якщо $D_5 = D_6 = 1$, то одержимо $A_3 A_2 A_1 A_0 = 0111$, що відповідає сигналу $D_7 = 1$. Усунути вказаний недолік можна шляхом застосування пріоритетних шифраторів, які при декількох збуджених вхідних змінних, формують код однієї із них, наприклад, старшої. У цьому випадку

$$A_3 = D_9 \vee D_8,$$
$$A_2 = \left(\overline{D_9 \vee D_8}\right) \wedge \left(D_7 \vee D_6 \vee D_5 \vee D_4\right),$$
$$A_1 = \left(\overline{D_9 \vee D_8}\right) \wedge \left(\left(D_7 \vee D_6\right) \vee \left(\overline{D_5 \vee D_4}\right) \wedge \left(D_3 \vee D_2\right)\right),$$
$$A_0 = D_9 \vee \overline{D}_8 \wedge \left(D_7 \vee \overline{D}_6 \wedge \left(D_5 \vee \overline{D}_4 \wedge \left(D_3 \vee \overline{D}_2 \wedge D_1\right)\right)\right).$$

На рис. 8.6 наведено схему пріоритетного шифратора, реалізовану за допомогою пакету MAX+plus II та результати моделювання його роботи.

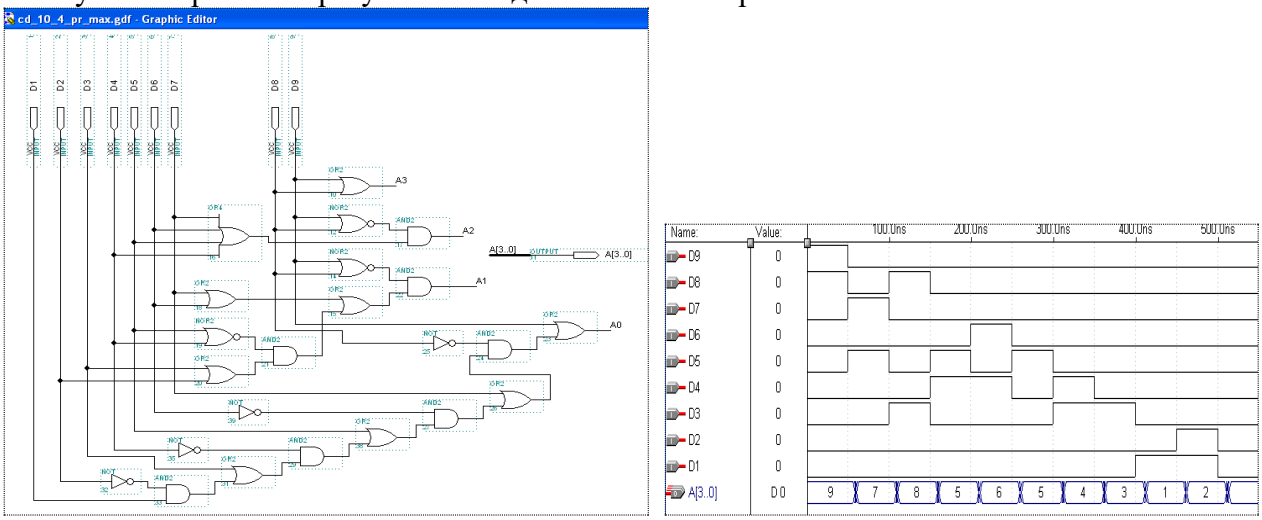


Рисунок 8.6

3. Завдання для самостійної роботи

3.1. Користуючись символьним редактором пакету MAX+plus II побудувати принципову схему (графічне зображення) дешифратора, наведеного на рис.8.2.

3.2. Користуючись пакетом MAX+plus II побудувати шифратор, заданий таблицею істинності 8.3 при $n = 2$ та дослідити його роботу.

3.3. Користуючись пакетом MAX+plus II реалізувати проект пріоритетного шифратора.

3.4. Користуючись пакетом MAX+plus II реалізувати проект мультиплексора з використанням символу внутрішнього дешифратора.

3.5. Користуючись пакетом MAX+plus II реалізувати пріоритетний шифратор клавіатури, який описаний нижче.

Лабораторна робота № 9

Тема: Типові вузли цифрової схемотехніки. Побудова мультиплексорів та демультиплексорів

Мета роботи: Набуття практичних навичок проектування мультиплексорів та демультиплексорів з використанням пакету MAX+PLUS II.

Теоретичні відомості

1. Мультиплексори

Цифровий мультиплексор або ***селектор даних*** – комбінаційна логічна схема, яка являє собою керований перемикач, який приймає декілька інформаційних сигналів, вибирає один із них і передає його на вихід (рис. 9.1).

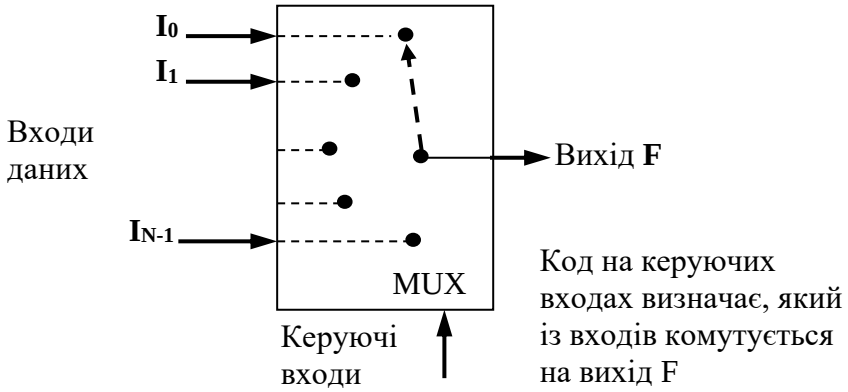


Рисунок 9.1 – Функціональна схема мультиплексора

Номер під'єднаного входу дорівнює числу (адресі), яке визначається комбінацією логічних значень на входах керування. Крім інформаційних входів і входів керування, схема мультиплексора може містити вхід дозволу, при подачі на який активного рівня, мультиплексор переходить в пасивний стан, при якому сигнал на виході зберігає постійне значення незалежно від значення інформаційних і керуючих сигналів. Число інформаційних входів у мультиплексора, як правило, є 2, 4, 8 або 16.

Мультиплексори застосовуються, наприклад, у мікропроцесорах для видачі на одні і ті ж виводи адреси і даних, що дає можливість істотно скоротити загальну кількість виводів мікросхеми; у мікропроцесорних системах керування мультиплексори встановлюються на віддалених об'єктах для можливості передачі інформації по одній лінії від декількох встановлених на них датчиків.

Мультиплексор з чотирма входами даних. Умовне графічне позначення мультиплексорів показане на рис. 9.2.

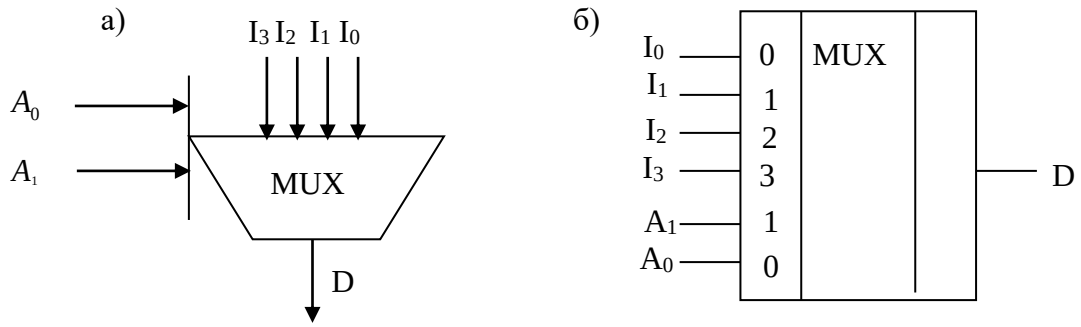


Рисунок 9.2 − Умовне позначення мультиплексорів:
а) на функціональних схемах; б) – на принципових схемах

Таблиця 9.1

A_0	A_1	F_0	F_1	F_2	F_3	D
0	0	1	0	0	0	F_0I_0
0	1	0	1	0	0	F_1I_1
1	0	0	0	1	0	F_2I_2
1	1	0	0	0	1	F_3I_3

Логіка роботи чотиривходового мультиплексора наведена в табл. 9.1, де A_0, A_1 – адресний код; F_0, F_1, F_2, F_3 – виходи внутрішнього дешифратора; I_0, I_1, I_2, I_3 – вхідна інформація; D – загальний інформаційний вихід.

На основі табл. 9.1 вираз для вихідної функції D можна подати з використанням виходів $F_0 - F_3$ внутрішнього дешифратора у вигляді

$$D = F_0 I_0 \vee F_1 I_1 \vee F_2 I_2 \vee F_3 I_3, \tag{9.1}$$

або з мінтермами адресного коду

$$D = \overline{A}_1 \overline{A}_0 I_0 \vee \overline{A}_1 A_0 I_1 \vee A_1 \overline{A}_0 I_2 \vee A_1 A_0 I_3. \tag{9.2}$$

1.1 Побудова мультиплексорів з використанням пакету MAX+PLUS II

Приклад 1. Побудувати мультиплексори з використанням пакету MAX+PLUS II.

Схеми мультиплексорів та результати моделювання, які відповідають рівнянням (9.1) та (9.2) показані на рис. 9.3 – 9.6.

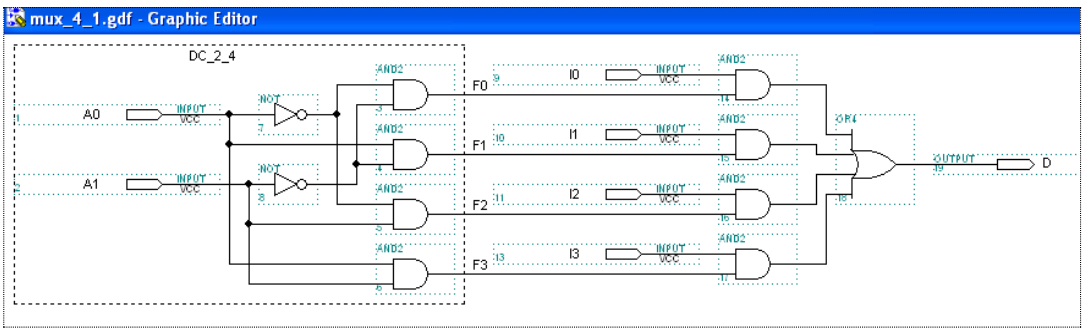


Рисунок 9.3

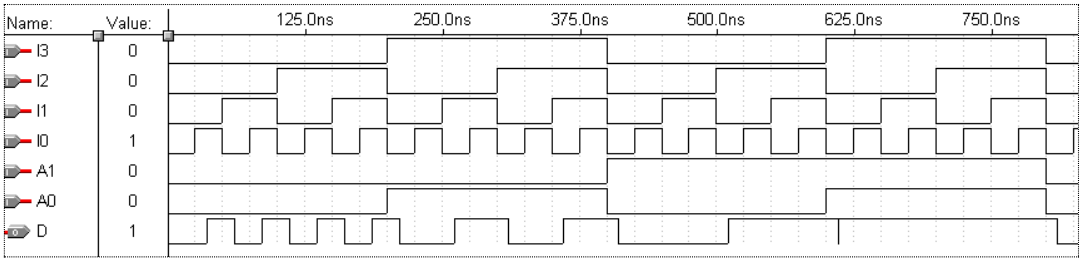


Рисунок 9.4

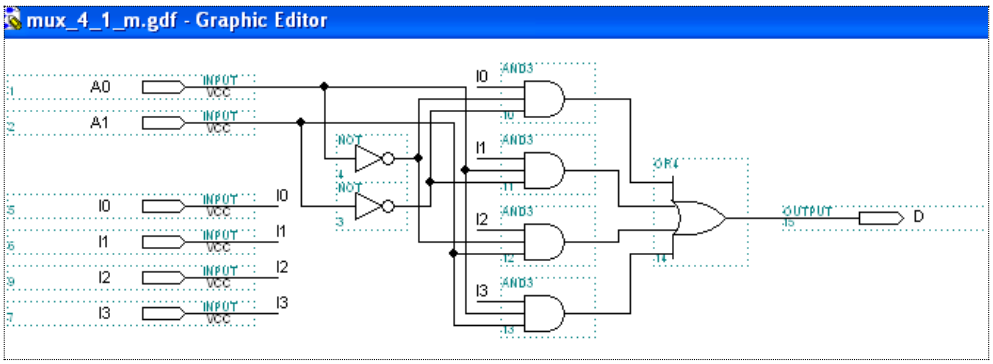


Рисунок 9.5

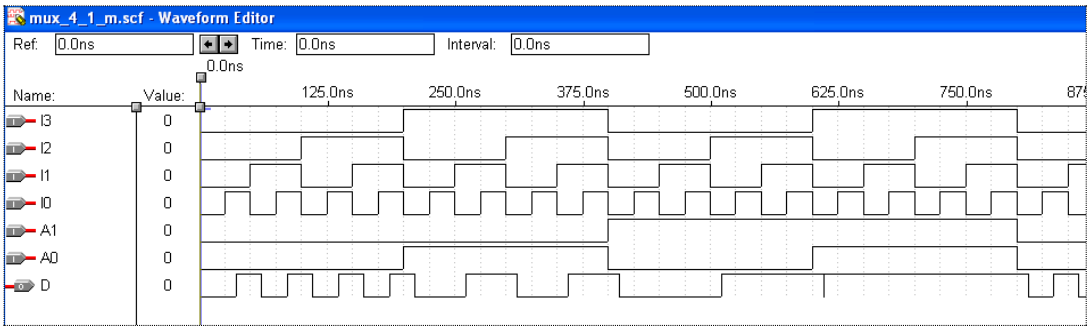


Рисунок 9.6

На рис. 9.7 наведено схему і символ внутрішнього дешифратора.

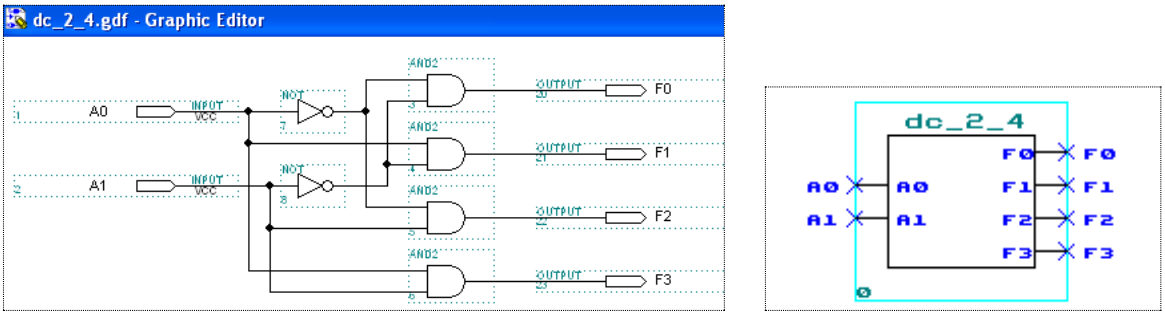


Рисунок 9.7

2. Демультиплексори

Демультиплексором називається функціональний вузол комп'ютера, який призначено для комутації (перемикання) сигналу з одного інформаційного входу D на один з n інформаційних виходів. Номер виходу, на який в кожний такт машинного часу передається значення вхідного сигналу, визначається адресним кодом $A_0, A_1, \ldots, A_m$. Адресні входи m та інформаційні виходи n пов'язані співвідношенням $n = 2^m$ або $m = \log_2 n$.

Демультиплексор виконує функцію, обернену функції мультиплексора.

В умовних графічних позначеннях функція демультиплексора позначається буквами *DMX*. На рис. 9.8 наведено умовні графічні позначення демультиплексорів: а) – на функціональних схемах; б) – на принципових схемах.

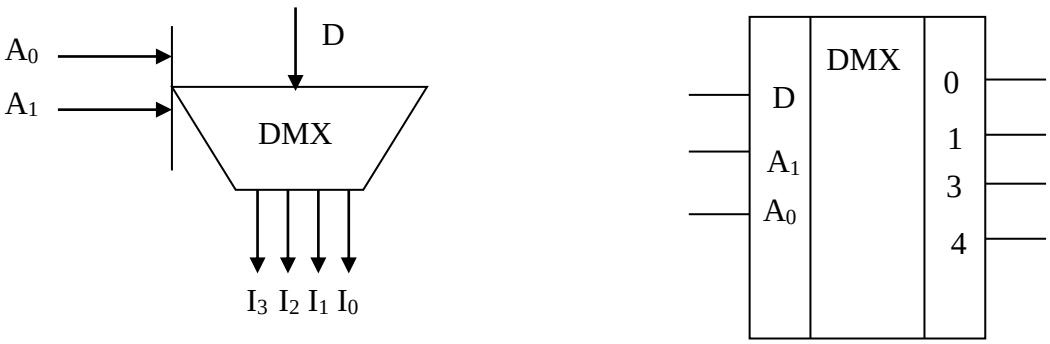


Рисунок 9.8 – Умовні графічні позначення демультиплексорів:
а) – на функціональних схемах; б) – на принципових схемах.

Демультиплексори використовуються для наступних операцій:

- комутації як окремих ліній, так і багаторозрядних шин;
- перетворення послідовного коду в паралельний;
- реалізації логічних функцій та ін

Логіка роботи двоадресного мультиплексора на мові мікрооперацій на ведена в табл. 3, де A_0, A_1 – адресний код; F_0, F_1, F_2, F_3 – виходи внутрішнього дешифратора адреси; I_0, I_1, I_2, I_3 – вихідна інформація; D – загальний інформаційний вхід.

Таблиця 9.2

A_1	A_0	F_0	F_1	F_2	F_3	I_0	I_1	I_2	I_3
0	0	1	0	0	0	F_0D	–	–	–
0	1	0	1	0	0	–	F_1D	–	–
1	0	0	0	1	0	–	–	F_2D	–
1	1	0	0	0	1	–	–	–	F_3D

За даними табл. 9.2 записуємо систему рівнянь для інформаційних виходів:

$$I_0 = F_0D = \overline{A}_1\overline{A}_0D,\ \ I_1 = F_1D = \overline{A}_1A_0D,\ \ I_2 = F_2D = A_1\overline{A}_0D,\ \ I_3 = F_3D = A_1A_0D. \qquad (9.3)$$

2.1. Побудова мультиплексорів з використанням пакету MAX+PLUS II

Приклад 2. На основі рівнянь (9.3) з використанням пакету MAX+PLUS побудовані схеми демультиплексорів із внутрішнім дешифратором (рис. 9.9) та з поєднанням адресних вхідних змінних на тривходових елементах I (рис. 9.11). Результати моделювання даних демультиплексорів наведено відповідно на рис. 9.10 та рис. 9.11.

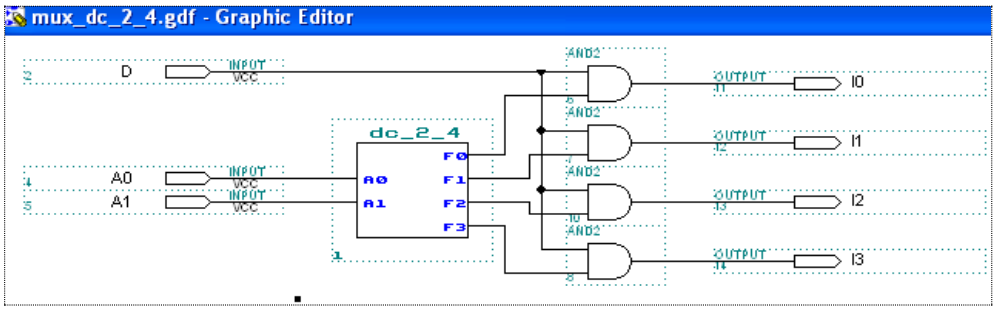


Рисунок 9.9

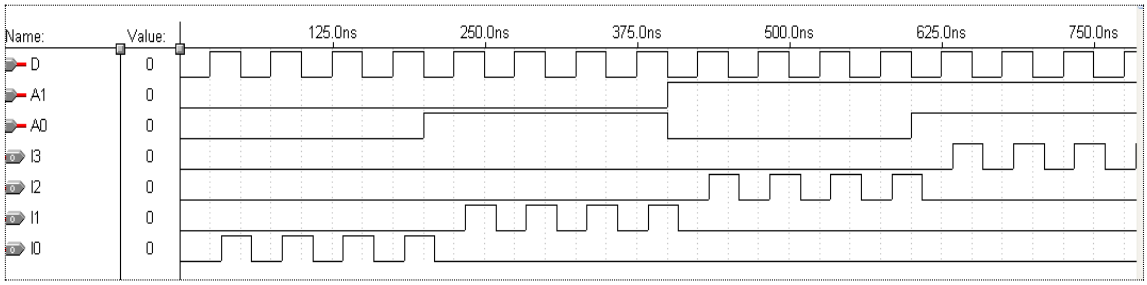


Рисунок 9.10

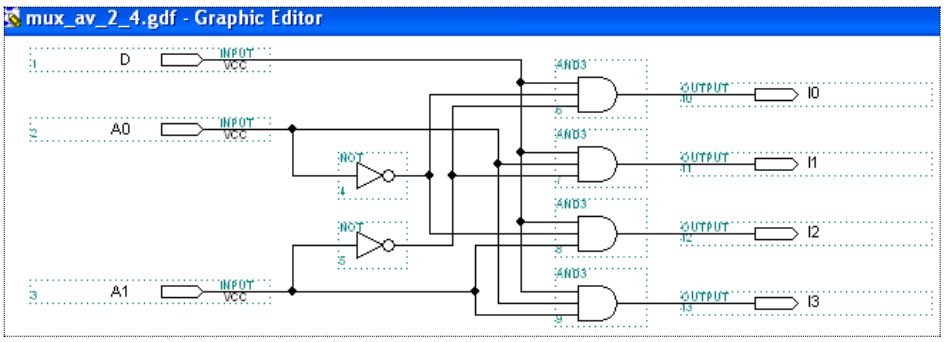


Рисунок 9.11

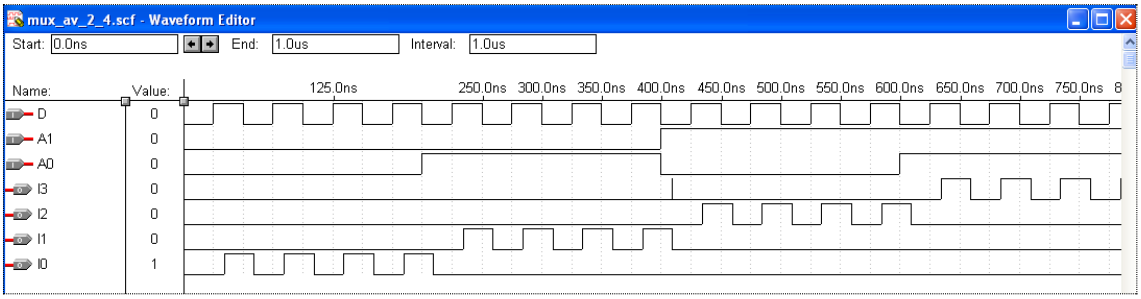


Рисунок 9.12

На рис. 9.13 наведено проект мультиплексора 4→1 з використанням символу внутрішнього дешифратора.

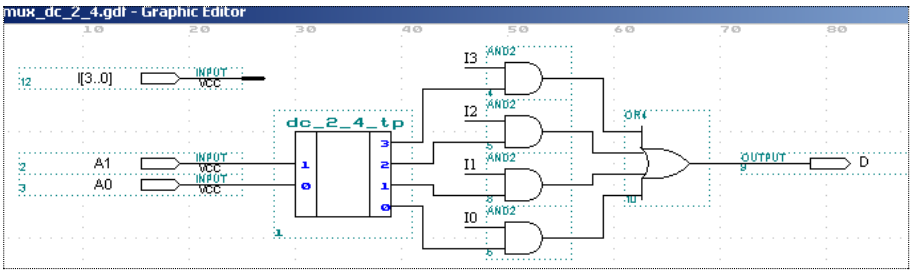


Рисунок 9.13 – Функціональна схема мультиплексора 4→1
з використанням символу внутрішнього дешифратора

Результати моделювання даного мультиплексора 4→1 з використанням символу внутрішнього дешифратора наведено на рис. 9.14:

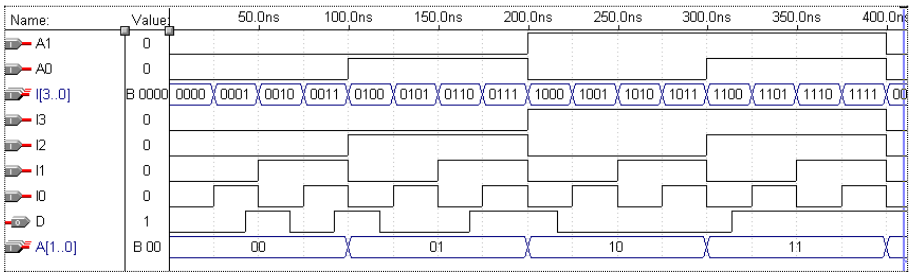


Рисунок 9.14 – Результати моделювання даного мультиплексора 4→1
з використанням символу внутрішнього дешифратор

3. Завдання для самостійної роботи

3.1. Користуючись символьним редактором пакетом MAX+plus II побудувати схеми мультиплексорів та навести результати моделювання, що відповідають рівнянням (9.1) та (9.2), які показані на рис. 9.3 – 9.6.

3.2. Користуючись пакетом MAX+plus II реалізувати проект мультиплексора 4→1 з використанням символу внутрішнього дешифратор.

Лабораторна робота № 10

Тема: Проектування цифрових автоматів з пам'яттю.

Мета роботи: навчитися проектувати цифрові автомати з пам'яттю за допомогою система проектування MAX+plus II.

Теоретичні відомості

1. Цифрові автомати з пам'яттю

Вузли і пристрої, які містять елементи пам'яті відносяться до класу автоматів з пам'яттю.

Цифровий автомат – це пристрій, який здійснює приймання, зберігання і перетворення дискретної інформації за деяким алгоритмом.

Абстрактний цифровий автомат визначається сукупністю п'яти об'єктів $\{X, S, Y, \varphi, \lambda\}$, де:

$X = \{x_i\}$, $i \in \overline{1,m}$ – множина вхідних сигналів автомата (вхідний алфавіт автомата);

$S = \{s_j\}$, $j \in \overline{1,n}$ – множина станів автомата (алфавіт станів автомата);

$Y = \{y_k\}$, $k \in \overline{1,l}$ – множина вихідних сигналів автомата (вихідний алфавіт автомата) ;

φ — функція переходів автомата, яка відображає (X×S)→S, тобто ставить у відповідність будь-якій парі елементів добутку множин (X×S) елемент множини S;

λ — функція виходів автомата, яка задає відображення (X×S)→Y або S →Y.

За способом формування функції виходів розрізняють наступні типи автоматів: автомат Мілі, автомат Мура (рис.1).

В абстрактному автоматі Мілі функція виходів λ задає відображення (X×S)→Y.

Автомат Мілі характеризується системою рівнянь:

$y(t) = \lambda[s(t), x(t)]; \quad s(t+1) = \varphi[s(t), x(t)]$.

Автомат Мура – системою рівнянь:

$y(t) = \lambda[s(t)]; \quad s(t+1) = \varphi[s(t), x(t)]$.

У практиці аналізу і синтезу автоматів використовують різні способи опису їх роботи. Найбільш поширеними є *табличний* і *графічний способи*.

За браком часу ми розглядатимемо лише опис автомата з використанням таблиць переходів та виходів (табличний спосіб). Стовбці (рядки) цих таблиць позначають символами з множини S, а рядки (стовбці) — символами з множини X. Для реалізації структурного синтезу цифрових автоматів скористаємось канонічним методом.

Згідно з канонічним методом синтез цифрових автоматів з пам'яттю можна розділити на наступні етапи:

1) задання таблиць переходів та виходів;

2) кодування вхідних, вихідних сигналів і внутрішніх станів автомата;

3) вибір елементів пам'яті автомата;

4) вибір типу автомату (Мілі, Муре);

5) побудова булевих функцій збудження і функцій виходів автомата (керуючих функцій);

6) побудова функціональної схеми автомата.

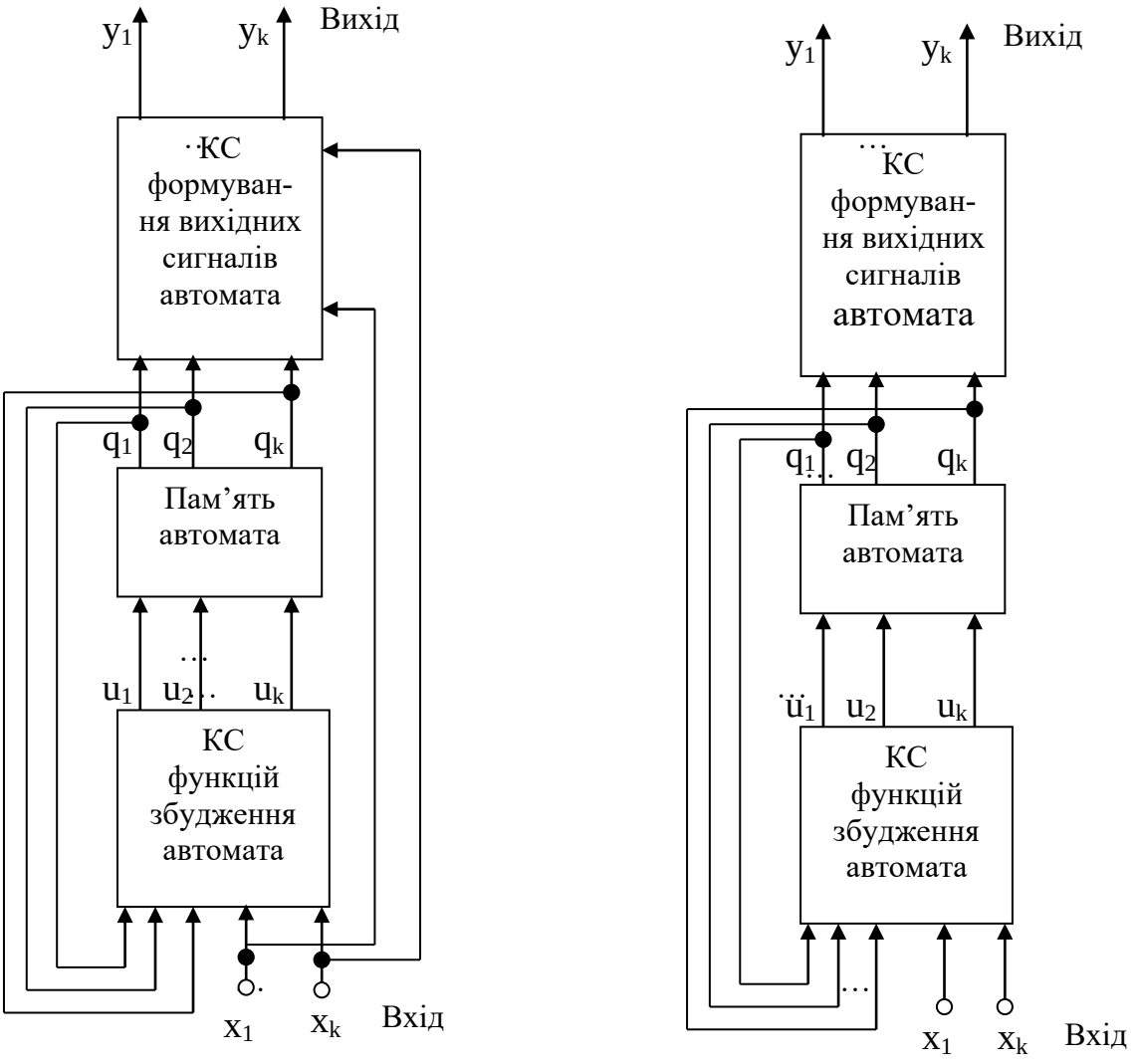


Рисунок 10.1 – Структурні схеми автоматів з пам'яттю

2. Канонічний метод структурного синтезу автомата

Приклад. Виконати структурний синтез цифрового автомата, заданого таблицями переходів і виходів (табл.10.1 і 10.2).

Таблиця 10.1 – Таблиця переходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_0	s_0	s_1
s_1	s_1	s_2
s_2	s_2	s_3
s_3	s_3	s_0

Таблиця 10.2 – Таблиця виходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_0	y_1	y_1
s_1	y_1	y_1
s_2	y_1	y_1
s_3	y_1	y_2

Синтез автомата виконати з використанням елементів пам'яті чотирьох типів тригерів: D, T, RS і JK тригерів.

2.1 Кодування

Процес заміни букв алфавітів X, S і Y цифрового автомата двійковими векторами називається кодуванням і може бути описаний таблицею (табл. 3, табл. 4, табл. 5). В лівій частині таблиці перераховуються всі букви (наприклад вхідного алфавіту), а в правій – двійкові вектори.

Таблиця 10.3

Стан автомата	*Код вхідних сигналів*
x_1	0
x_2	1

Таблиця 10.4

Стан автомата	*Код стану* q_1q_2
s_0	00
s_1	01
s_2	10
s_3	11

Таблиця 10.5

Вихідні сигнали	*Код вихідних сигналів*
y_1	0
y_2	1

Таблиця переходів і виходів після кодування має вигляд:

Таблиця 10.6 – Таблиця переходів

Стан автомата	*Вхідні сигнали*	
	$x=0$	$x=1$
00	00	01
01	01	10
10	10	11
11	11	00
q_2q_1		

Таблиця 10.7 – Таблиця виходів

Стан автомата	*Вхідні сигнали*	
	$x=0$	$x=1$
00	0	0
01	0	0
10	0	0
11	0	1
q_2q_1	y	y

2.2 Вибір елементів пам’яті автомата

Як уже було сказано, в якості елементів пам’яті структурного автомата використовують D –, T –, RS – і JK – тригери.

У подальшому (для зручності) при структурному синтезі автомата з будемо наводити таблицю переходів відповідного тригера, вихідну таблицю станів автомата та будуватимемо відповідну таблицю переходів.

Що стосується таблиці виходів, то вона є однаковою для всіх типів тригерів, оскільки вихідні сигнали залежать тільки від вхідних сигналів і початкових станів автоматів. Логічне рівняння булевої функції виходу y автомата, що синтезуємо, одержуються з табл. 10.7 має вигляд $y = q_2q_1x$.

3. Синтез автомата з використанням D –тригера

Нижче наводимо таблицю переходів D-тригера (табл.10.8) і таблицю переходів автомата (табл. 10.9). Легко переконатись, що таблиця збудження автомата для D-тригера співпадає з таблицею переходів автомата.

Таблиця 10.8

Q_t	Перехід	Q_{t+1}
0	$\xrightarrow{D=0}$	0
0	$\xrightarrow{D=1}$	1
1	$\xrightarrow{D=0}$	0
1	$\xrightarrow{D=1}$	1

Таблиця 10.9

Стан автомата	*Вхідні сигнали*	
	$x=0$	$x=1$
00	00	01
01	01	10
10	10	11
11	11	00
q_2q_1	D_2D_1	D_2D_1

На основі таблиці збудження (табл.8) складаємо рівняння для побудови комбінаційної схеми збудження цифрового автомата:

$$D_1 = \bar{q}_2 q_1 \bar{x} + q_2 q_1 \bar{x} + \bar{q}_2 \bar{q}_1 x + q_2 \bar{q}_1 x = q_1 \bar{x} + \bar{q}_1 x = q_1 \oplus x;$$

$$D_2 = q_2 \bar{q}_1 \bar{x} + q_2 q_1 \bar{x} + \bar{q}_2 q_1 x + q_2 \bar{q}_1 x = q_2 \bar{x} + \bar{q}_2 q_1 x + q_2 \bar{q}_1 x = q_2 \bar{x} + (q_1 + q_2) x.$$

Функціональна схема автомата Мілі, реалізована на основі D-тригерів та результати її моделювання наведено на рис. 10.2

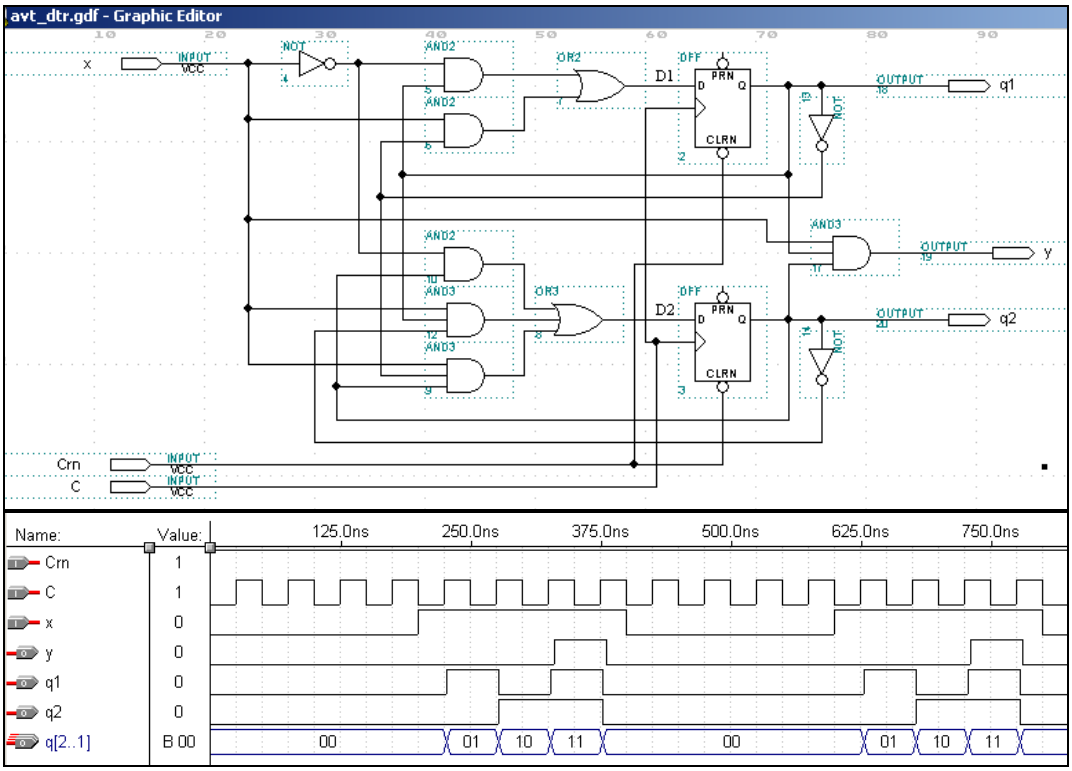


Рисунок 10.2

4. Особливості синтезу автоматів на основі T-, RS-, JK-тригерів

Відмітимо, що синтез на базі вказаних типів тригерів здійснюється аналогічно виконаному синтезу на основі D–тригерів. Зокрема, вибирається таблиця переходів відповідного тригера, кодуються вхідні, вихідні і внутрішні стани автомата та будуються кодовані таблиці переходів і виходів структурного автомата.

4.1 Синтез автомата з використанням T-тригера

Таблиця переходів T-тригера

Q_t	Перехід	Q_{t+1}
0	$\xrightarrow{T=0}$	0
0	$\xrightarrow{T=1}$	1
1	$\xrightarrow{T=1}$	0
1	$\xrightarrow{T=0}$	1

Таблиця переходів автомата

Стан автомата	*Вхідні сигнали*	
	$x=0$	$x=1$
00	00	01
01	01	10
10	10	11
11	11	00
q_2q_1	T_2T_1	T_2T_1

Таблиця збудження автомата

	Стан автомата	*Вхідні сигнали*	
		$x=0$	$x=1$
	00	00	01
$\Rightarrow$	01	00	11
	10	00	01
	11	00	11
	q_2q_1	T_2T_1	T_2T_1

На основі одержаної таблиці збудження складаємо логічні формули для побудови комбінаційної схеми збудження цифрового автомата:

$$T_1=\overline{q}_2\overline{q}_1x+\overline{q}_2q_1x+q_2\overline{q}_1x+q_2q_1x=\overline{q}_2x+q_2x=x;\quad T_2=\overline{q}_2q_1x+q_2q_1x=q_1x.$$

Функціональна схема автомата Мілі, побудована на основі Т-тригера та результати її моделювання наведено на рис.10.3.

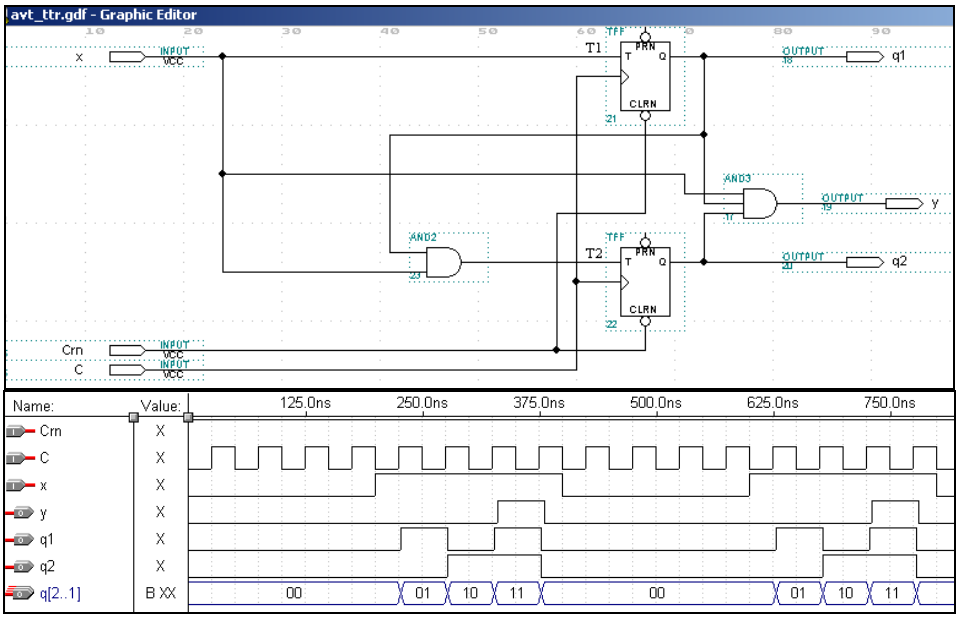


Рисунок 10.3

4.2 Синтез автомата з використанням RS-тригера

Таблиця переходів RS-тригера

Q_t	Перехід	Q_{t+1}
0	$R=^*;\ S=0$	0
0	$R=0;\ S=1$	1
1	$R=1;\ S=0$	0
1	$R=0;\ S=^*$	1

Таблиця переходів автомата

Стан автомата	*Вхідні сигнали*	
	$x=0$	$x=1$
00	00	01
01	01	10
10	10	11
11	11	00
q_2q_1		

Таблиця збудження автомата

	Стан автомата	*Вхідні сигнали*			
		$x=0$		$x=1$	
	00	* 0	* 0	* 0	0 1
⇒	01	* 0	0 *	0 1	1 0
	10	0 *	* 0	0 *	0 1
	11	0 *	0 *	1 0	1 0
	q_2q_1	$R_2\ S_2$	$R_1\ S_1$	$R_2\ S_2$	$R_1\ S_1$

На основі одержаної таблиці збудження складаємо логічні формули для побудови комбінаційної схеми збудження цифрового автомата:

$R_1=\bar{q}_2q_1x+q_2q_1x=q_1x;\ S_1=\bar{q}_2\bar{q}_1x+q_2\bar{q}_1x=\bar{q}_1x;\ R_2=q_2q_1x;\ S_2=\bar{q}_2q_1x.$

Функціональна схема автомата Мілі, побудована на основі RS-тригерів та результати її моделювання наведено на рис. 10.4.

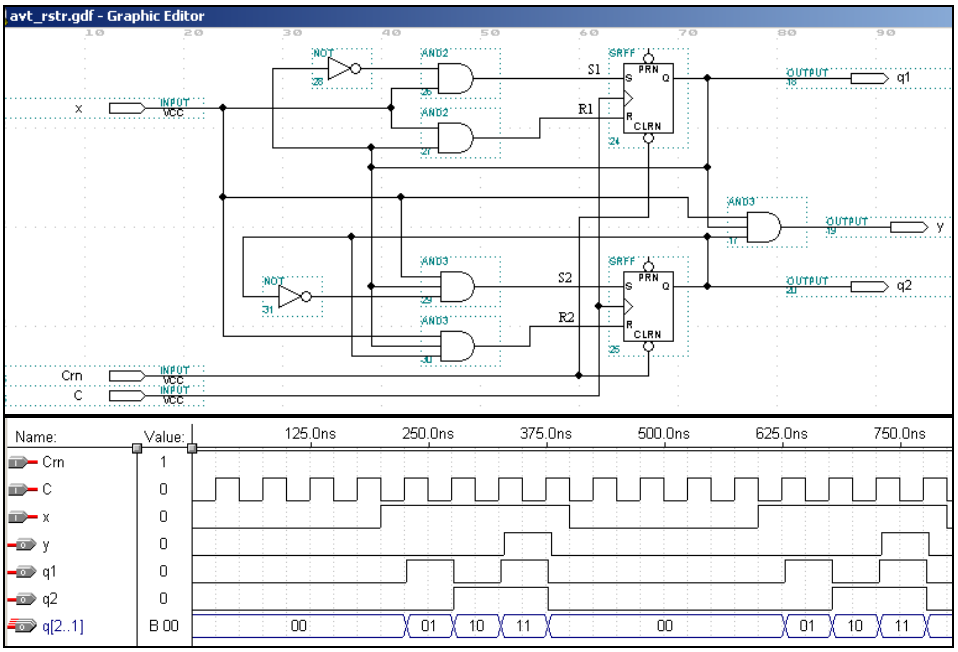


Рисунок 10.4

4.3 Синтез автомата з використанням JK-тригера

Таблиця переходів JK-тригера

Q_t	Перехід	Q_{t+1}
0	$J=0;\ K=*$	0
0	$J=1;\ K=*$	1
1	$J=*;\ K=1$	0
1	$J=*;\ K=0$	1

Таблиця переходів автомата

Стан автомата	*Вхідні сигнали*	
	$x=0$	$x=1$
00	00	01
01	01	10
10	10	11
11	11	00
q_2q_1		

Таблиця збудження автомата

	Стан автомата	*Вхідні сигнали*			
		$x=0$		$x=1$	
	00	0 *	0 *	0 *	1 *
$\Rightarrow$	01	0 *	* 0	1 *	* 1
	10	* 0	0 *	* 0	1 *
	11	* 0	* 0	* 1	* 1
	q_2q_1	J_2K_2	J_1K_1	J_2K_2	J_1K_1

На основі одержаної таблиці збудження складаємо логічні формули для побудови комбінаційної схеми збудження цифрового автомата:

$J_1=\bar{q}_2\bar{q}_1x+q_2\bar{q}_1x=\bar{q}_1x;\ K_1=\bar{q}_2q_1x+q_2q_1x=q_1x;\ J_2=\bar{q}_2q_1x;\ K_2=q_2q_1x.$

Функціональна схема автомата Мілі, побудована на основі JK-тригерів та результати її моделювання наведено на рис. 10.5.

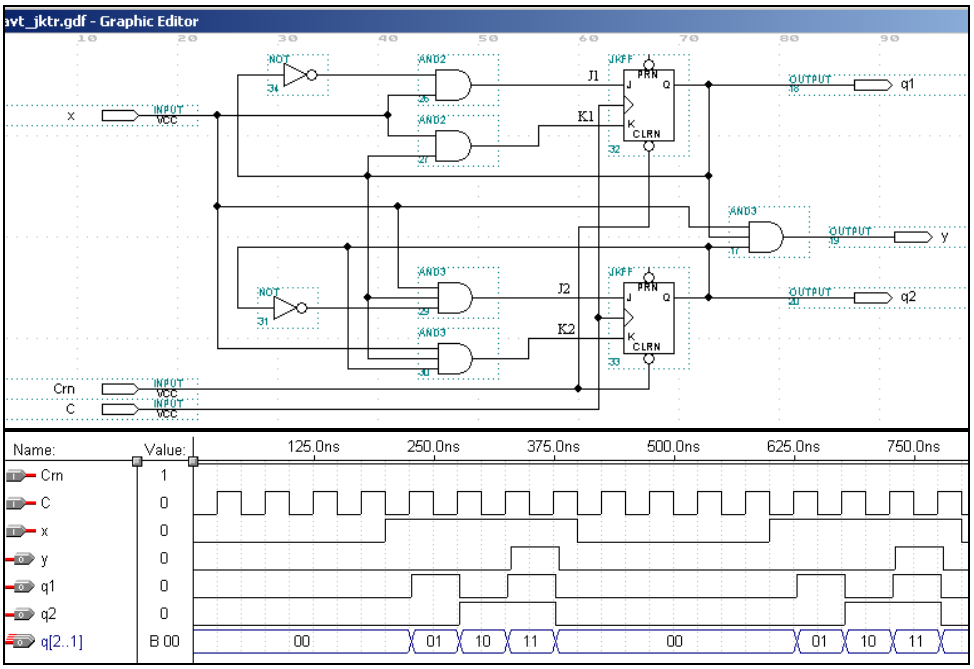


Рисунок 10.5

5. Порядок виконання роботи

5.1. Замінюємо букви алфавітів S, Y , X цифрового автомата двійковими векторами.

5.2. Складаємо рівняння для побудови комбінаційної схеми збудження цифрового автомата

5.3. Складаємо рівняння для побудови комбінаційної схеми формування вихідних сигналів автомата.

5.4. Будуємо функціональну схему цифрового автомата.

5.5. Перевіряємо роботу цифрового автомата.

6. Зміст звіту

6.1. Тема і мета роботи.
6.2. Вихідні дані для виконання роботи.
6.3. Результати виконання пунктів 5.1 – 5.5
6.4. Функціональні схеми роботи цифрового автомата з пам'яттю та часові діаграми.
6.5. Висновки.

7. Індивідуальні завдання для лабораторної роботи

Спроектувати цифровий автомат Мілі згідно заданих таблиць переходів і виходів. Комбінаційну схему збудження і комбінаційну схему формування вихідних сигналів реалізувати в заданому базисі.

Варіант № 1

Таблиця переходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_1	s_1
s_2	s_2	s_2
s_3	s_3	s_2

Таблиця виходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_2	y_3
s_2	y_2	y_4
s_3	y_1	y_4

Тип тригера – J-K. Базис –І-НЕ.

Варіант № 2

Таблиця переходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_1	s_1
s_2	s_2	s_1
s_3	s_3	s_2

Таблиця виходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_1	y_3
s_2	y_2	y_4
s_3	y_1	y_2

Тип тригера – J-K. Базис –АБО-НЕ.

Варіант № 3

Таблиця переходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_2	s_3
s_2	s_1	s_1
s_3	s_3	s_2

Таблиця виходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_3	y_1
s_2	y_2	y_4
s_3	y_1	y_2

Тип тригера – R-S. Базис –І-НЕ.

Варіант № 4

Таблиця переходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_2	s_2
s_2	s_3	s_1
s_3	s_1	s_2

Таблиця виходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_1	y_3
s_2	y_2	y_4
s_3	y_1	y_2

Тип тригера – R-S. Базис –АБО-НЕ.

Варіант № 5

Таблиця переходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_3	s_3
s_2	s_2	s_1
s_3	s_3	s_2

Таблиця виходів

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_1	y_2
s_2	y_3	y_4
s_3	y_2	y_2

Тип тригера – J-K. Базис –І-НЕ.

Варіант № 6

Таблиця переходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_1	s_2
s_2	s_2	s_1
s_3	s_3	s_2

Таблиця виходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_4	y_3
s_2	y_2	y_4
s_3	y_1	y_2

Тип тригера – J-K. Базис –АБО-НЕ.

Варіант № 7

Таблиця переходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_1	s_3
s_2	s_2	s_1
s_3	s_1	s_2

Таблиця виходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_1	y_3
s_2	y_2	y_4
s_3	y_2	y_2

Тип тригера – R-S. Базис –І-НЕ.

Варіант № 8

Таблиця переходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_3	s_3
s_2	s_2	s_1
s_3	s_3	s_2

Таблиця виходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_4	y_3
s_2	y_2	y_4
s_3	y_1	y_1

Тип тригера – R-S. Базис –АБО-НЕ.

Варіант № 9

Таблиця переходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_2	s_2
s_2	s_1	s_1
s_3	s_3	s_2

Таблиця виходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_1	y_3
s_2	y_4	y_4
s_3	y_1	y_2

Тип тригера – J-K. Базис –І-НЕ.

Варіант № 10

Таблиця переходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_2	s_1
s_2	s_3	s_1
s_3	s_3	s_2

Таблиця виходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_2	y_3
s_2	y_2	y_4
s_3	y_1	y_2

Тип тригера – J-K. Базис –АБО-НЕ.

Варіант № 11

Таблиця переходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_2	s_2
s_2	s_2	s_1
s_3	s_3	s_2

Таблиця виходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_4	y_3
s_2	y_2	y_4
s_3	y_1	y_2

Тип тригера – J-K. Базис –І-НЕ.

Варіант № 12

Таблиця переходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_1	s_2
s_2	s_2	s_1
s_3	s_3	s_2

Таблиця виходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_1	y_3
s_2	y_2	y_4
s_3	y_2	y_2

Тип тригера – J-K. Базис –АБО-НЕ.

Варіант № 13

Таблиця переходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_2	s_1
s_2	s_1	s_3
s_3	s_3	s_2

Таблиця виходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_4	y_3
s_2	y_2	y_4
s_3	y_1	y_3

Тип тригера – R-S. Базис –І-НЕ.

Варіант № 14

Таблиця переходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_2	s_1
s_2	s_2	s_1
s_3	s_1	s_2

Таблиця виходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_1	y_3
s_2	y_1	y_4
s_3	y_2	y_2

Тип тригера – R-S. Базис –АБО-НЕ.

Варіант № 15

Таблиця переходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_1	s_1
s_2	s_1	s_2
s_3	s_3	s_2

Таблиця виходів.

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_1	y_3
s_2	y_2	y_4

Тип тригера – J-K. Базис –І-НЕ.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Антонов А.П. Язык описания цифровых устройств AlteraHDL. Практический курс. - Г.: ИП Радиософт, 2001. - 224 с.
2. Армстронг Дж. Р. Моделирование цифровых систем языком VHDL.: Пер. с англ./М.: Мир, 1992. - 175 с.
3. Бибило П.Н. Основы языка VHDL. - Г.: ООО Издательство "Солон-Р", 2000. - 200 с.
4. К.Г. Самофалов, А.М. Романкевич, В.Н. Валуйський, Ю.С. Канівський, М.М. Піневич. Прикладная теория цифровых автоматов: Учеб. для ВУЗов – К.: Высш. шк., 1987. – 375 с.
5. Комп'ютерна схемотехніка. Проектування типових вузлів комп'ютерних систем: Метод. вказівки до викон. лабораторн. робіт. / Уклад.: І.А Дичка., В.І. Жабін, В.П.Тарасенко. К.: НТУУ «КПІ», 2006. – 92 с.
6. Рябенький В.М., Ушкаренко О.О. MAX+PLUS II. Основи проектування цифрових пристроїв на ПЛІС. – К.: « Корнійчук», 2004. – 253 с.
7. Король І.Ю. Проектування комбінаційних схем в програмному комплексі MAX+PLUS II (конспект лекцій).

ЗМІСТ