

# Лабораторна робота № 1. Практична частина

## ВІВЧЕННЯ СИСТЕМИ АВТОМАТИЗАЦІЇ ПРОЕКТУВАННЯ *QUARTUS II*. СИНТЕЗ КОМБІНАЦІЙНИХ СХЕМ.

### Ціль роботи:

- Вивчення системи автоматизації проектування *Quartus II*. Отримання навиків створення проекту, вводу проекту в схемотехнічному режимі, роботи в графічному редакторі, ~~створення мегафункцій~~.
- Вивчення особливостей функціональної побудови суматорів. Розроблення функціональної схеми суматора в САПР *Quartus II*.

### Теоретичні відомості

Програмне середовище *Quartus® II* компанії *Altera®* є повною мультиплатформеною системою для автоматизації проектування (САПР), що містить набір інструментів для проектування цифрових пристроїв та систем на програмовному кристалі (*SOPC*). Середовище *Quartus II* включає в себе всі утиліти, необхідні для роботи з мікросхемами *FPGA* і *CPLD*. На рисунку 1.1 показані основні етапи проектування (*Design Flow*) в середовищі *Quartus II*.

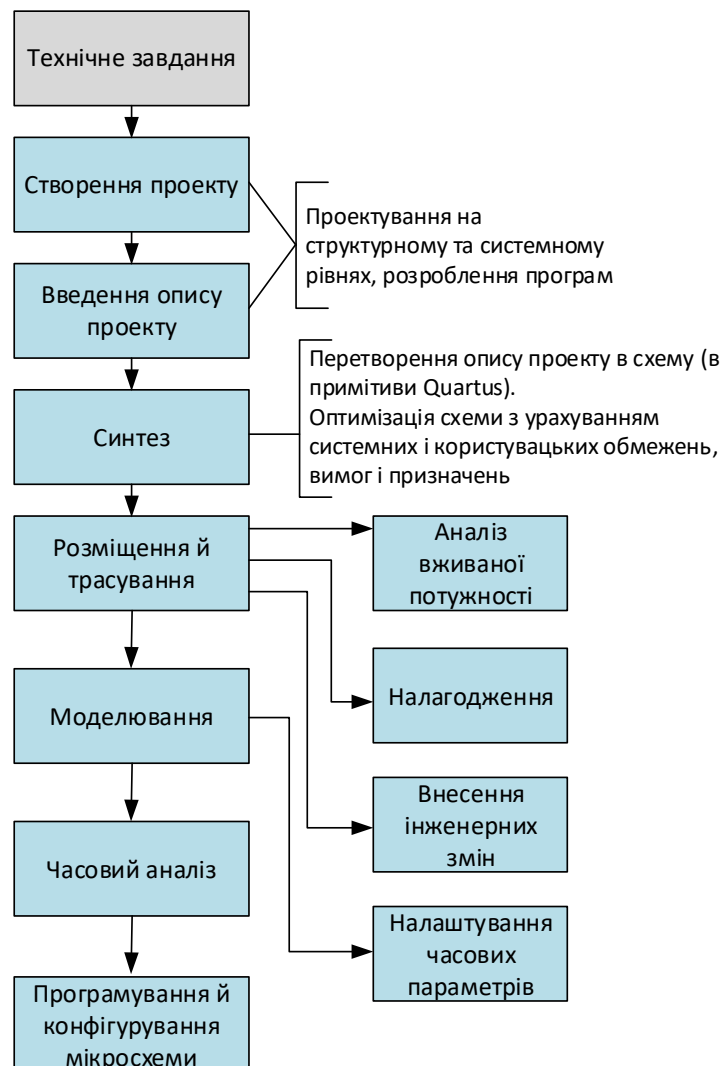


Рис. 1.1. Етапи стандартного процесу проектування в САПР *Quartus II*

Стандартний процес проектування включає наступні етапи.

## I. Введення опису проекту

1. Створення нового проекту, вибір сімейства й типу мікросхеми. Використовується майстер створення проектів, який запускається командою **New Project Wizard** із меню **File**.

2. Введення опису проекту в САПР. Можна використовувати текстовий або графічний опис файлів проекту. За обома способами необхідно створити вихідні файли проекту, в тому числі й файл верхнього рівня ієрархії проекту (*Top Level*). За першим способом, вихідні файли створюються на мовах проектування апаратури *Verilog HDL*, *VHDL*, або *Altera Hardware Description Language (AHDL)* за допомогою текстового редактора (*Text Editor*). За використання графічного вводу блок-схема проекту створюється в графічному редакторі (*Block Editor*) з використанням уніфікованих символів, які представляють собою окремі блоки схеми, що описуються іншими вихідними файлами, та примітиви. В графічному режимі вводу проекту за допомогою майстра **MegaWizard® Plug-In Manager** (меню **Tools**) можна створювати різні мегафункції й IP-ядера і включати їх в файли проекту.

3. Вказання початкових налаштувань проекту. Використовується редактор призначень (**Assignment Editor**), діалогового вікна **Settings** (меню **Assignments**), редактор топології (**Floorplan Editor**), та/або технологія фіксованих логічних блоків (*LogicLock™*).

4. Проект може бути створено за допомогою спеціальних засобів для створення систем-на-кристалі (*SOPC Builder*) або для створення систем цифрової обробки сигналів (ЦОС) (*DSP Builder*). Програмні файли для процесорного ядра *Nios®* створюються за допомогою редактора програмного забезпечення (*Software Builder*).

## II. Синтез проекту

5. Синтез проекта. Використовується модуль аналізу й синтезу (**Analysis & Synthesis**).

6. Функціональне моделювання проекту. Використовується симулятор (**Simulator**) и команда **Generate Functional Simulation Netlist**.

7. Розміщення і трасування проекту. Використовується модуль трасування (**Fitter**).

8. Попередній аналіз споживаної потужності. Використовується утиліта **PowerPlay Power Analyzer**.

9. Аналіз часових затримок. Використовується утиліта аналізатор часових затримок (**Timing Analyzer**).

## III. Моделювання проекту

10. Моделювання проекту з урахуванням часових затримок. Використовується симулятор (**Simulator**).

11. Покращення часових характеристик проекту. Здійснюється повторний фізичний синтез проекту, налаштування в діалоговому вікні **Settings** та в редакторі призначень, фіксовані логічні блоки.

## IV. Програмування мікросхеми

12. Створення файлу для програмування пристрою на мікросхемі. Використовується модуль асемблера (**Assembler**).

13. Програмування мікросхеми за допомогою утиліти програматора (**Programmer**) й устаткування *Altera*; або перетворення формату файла для програмування.

## V. Внутрішньокристалъне налагодження проекту

14. Налагодження проекту на мікросхемі. Використовується убудований логічний аналізатор (*SignalTap® II Logic Analyzer*), генератор контрольних крапок (*SignalProbe™*).

Проект є основина одиниця в САПР *Quartus II*. Проект створюється за допомогою майстра створення нового проекту **New Project Wizard**, що знаходиться в меню **File**. Майстер дозволяє задати робочу директорію, де будуть зберігатися файли проекту, призначити ім'я проекту і ім'я файлу верхнього рівня ієрархії проекту. До проекту можна підключити інші вихідні файли, користувацькі бібліотеки, САПР сторонніх компаній, що використовуються, обрати сімейство або тип мікросхем для реалізації проекту або використати автоматичний підбір мікросхем компілятором *Quartus II*.

### Порядок виконання роботи

#### Створення нового проекту

**1.** Запустіть САПР *Quartus II*. Для виконання циклу лабораторних робіт використовується САПР *Quartus II Version 9.1 Build 222 10/21/2009 SJ Full Version*.

Запуск пакету виконується або з використанням іконки, розташованої на робочому столі комп'ютера, або з меню ПУСК: **Пуск** → **Всі програми** → **Altera** → **Quartus II Web Edition Full**. Для запуску з командного рядка наберіть **Quartus** і натисніть **Enter**. Після запуску з'явиться на екрані головне вікно пакету, яке має декілька робочих зон (рис. 1.2).

**2.** Створіть новий проект за допомогою майстра **New Project Wizard**.

Послідовність створення нового проекту:

**2.1** Створіть в робочій директорії жорсткого диска папку з назвою проекту, наприклад **CL\_Project\_v1 \*\*\***, інакше майстер запитає про створення нової папки на наступному кроці;

**\*\*\*** Замість **Project** введіть прізвище студента для подальшої ідентифікації проектів.

**2.2.** В меню **File** оберіть команду **New Project Wizard** (рис. 1.3). Відкриється вікно майстра. При першому запуску відображається вікно заставки (**Introduction**), в якому відображена послідовність дій, що супроводжує відкриття нового проекту. Якщо на екрані з'явиться сторінка **Introduction**, натисніть кнопку **Next**.

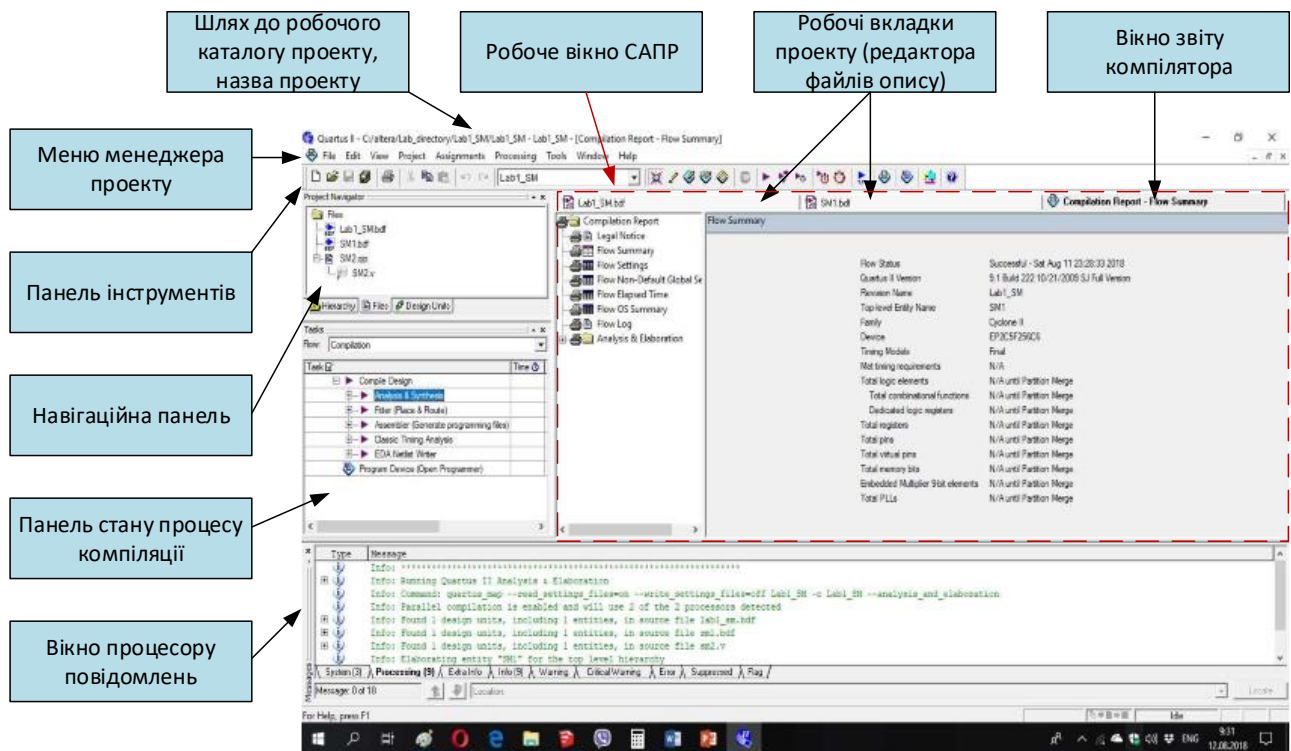


Рис. 1.2. Головне вікно середовища *Quartus II*.

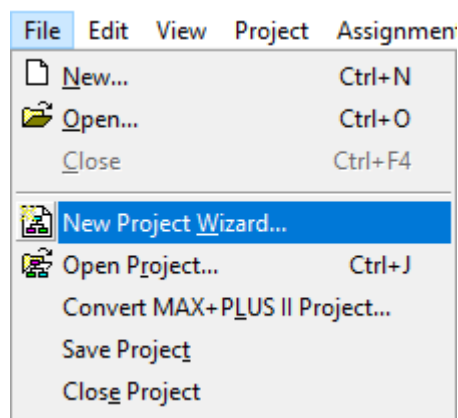



Рис. 1.3. Визивання майстра **New Project Wizard**

**2.3.** На першій сторінці майстра створення проекту (рис. 1.4) введіть наступну інформацію в текстові поля (приклад заповнення наведено в табл. 1):

Таблиця 1.1

Приклад для налаштування першої сторінки  
майстра **New Project Wizard**

Робочий каталог проекту	..\Lab_AC_Directory\ <b>CL_Project_v1</b>
Ім'я проекту	<b>CL_Project_v1</b>
Ім'я об'єкту верхнього рівня розробки (файлу проекту)	<b>CL_Project_v1</b>

- Назва робочого каталогу проекту. Каталог буде містити всі робочі файли проекту та інші файли, що пов'язані з даним проектом. Якщо каталог раніше не був створений, *Quartus II* створить його автоматично. Якщо каталог вже був створений необхідно вказати шлях до нього, натиснувши на піктограму  праворуч від текстового поля (рис. 1.4). Введене ім'я автоматично відображається в інших текстових полях вікна;
- Ім'я проекту.

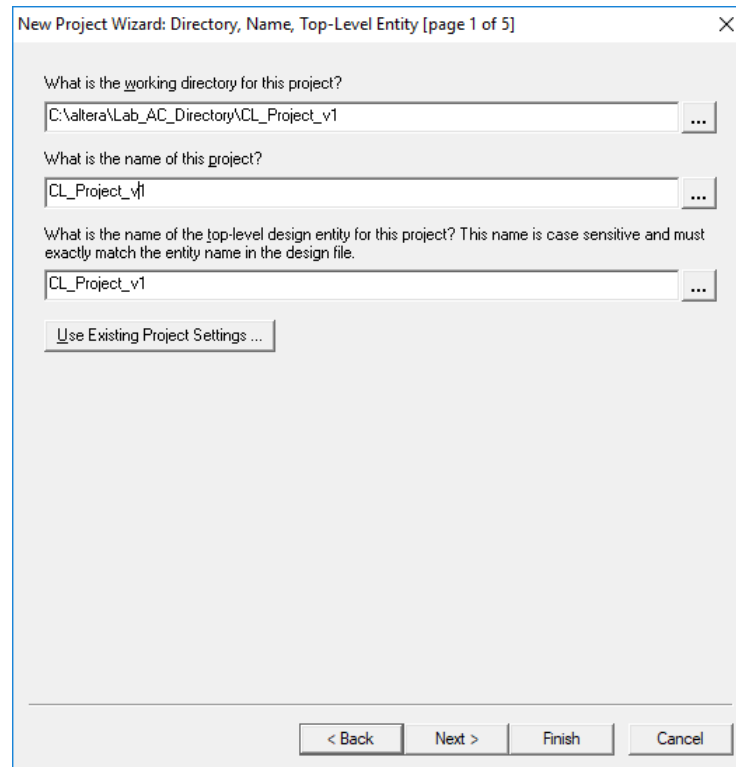



Рис. 1.4. Приклад налаштування першої сторінки майстра **New Project Wizard**

• Ім'я файлу верхнього рівня проекту. *Quartus II* автоматично створює установки компіляції та моделювання для файлу верхнього рівня проекту (**Top-Level Design Entity**) з зазначеним ім'ям. Задане ім'я проекту автоматично присвоюється головному файлу проекту в відповідному текстовому полі вікна. Припустимо задавати імена, відмінні від імені файлу проекту, але не рекомендується. Після створення проекту Ви зможете додати інші файли проекту і створити для них установки компіляції та моделювання за допомогою меню **Processing**.

Для переходу в наступне вікно натисніть кнопку **Next**;

**2.4.** Друге вікно майстра створення проекту (рис. 1.5) призначене для підключення до нового проекту раніше створених файлів. Для цього натисніть піктограму  і оберіть файл, який Ви хочете додати до Вашого проекту. Натисніть кнопку **Open**, після чого кнопку **Add**. Якщо Ви хочете підключити до проекту декілька файлів, виберіть їх імена і натисніть кнопку **Add All**.

Файл верхнього рівня (файл проекту) не треба додавати до проекту, цей файл автоматично додається до проекту майстром створення проекту. Додавання

файлів і каталогів на цьому кроці виконується, якщо файли не знаходяться в робочій директорії проекту, були створені раніш, наприклад у складі інших проектів. Якщо Ви не маєте файлів проекту в інших каталогах або файлів, ім'я яких не співпадають з іменем проекту, то додавати файли у цьому вікні не потрібно.

Слід звернути увагу, що до проекту додаються лише шляхи до приєднаних файлів, фізично файли знаходяться в своїх вихідних директоріях. Для архівування таких проектів слід використовувати спеціальні засоби для компіляції пакету Quartus II.

Окрім файлів на другій сторінці можна додати бібліотеки спеціалізованих функцій, вказавши шлях до них. Це блоки, або функції, які були розроблені у складі інших проектів або сторонніми розробниками. Для цього слід натиснути кнопку **User Library**.

Під час виконання даної лабораторної роботи непотрібно підключати додаткові файли і бібліотеки, таким чином, для завершення роботи з другим вікном натисніть кнопку **Next**.

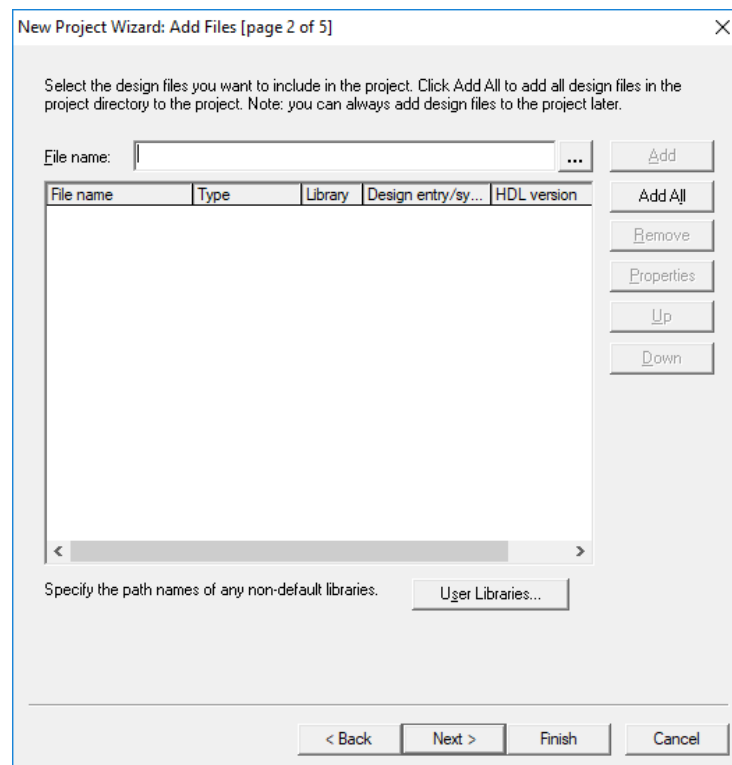


Рис.1.5. Налаштування другої сторінки майстра **New Project Wizard**

**2.5.** Третє вікно призначене для вибору сімейства і типу мікросхеми ПЛІС, які будуть використані під час компіляції проекту (рис.1.6).

Виберіть сімейство (**Family**) *Cyclone II*.

Для обмеження списку доступних мікросхем, можна використати спеціальні фільтри, якщо Вам відомі певні параметри мікросхеми. Наприклад, у частині вікна, що розміщена праворуч зверху, в розділі **Show in 'Available device' list**, встановіть наступні значення: в рядку **Package** оберіть **FBGA**, в рядку **Pin count** (кількість пінів) – **256**, **Speed grade** (швидкодія) – **Fastest**. У вікні **Available devices**

оберіть мікросхему **EP2C5F256C6**. Натисніть **Next** для переходу на наступну сторінку.

Select the family and device you want to target for compilation.

Device family:  
Family: Cyclone II  
Devices: All

Target device:  
☐ Auto device selected by the Filter  
☒ Specific device selected in 'Available devices' list

Show in 'Available device' list:  
Package: FBGA  
Pin count: 256  
Speed grade: Fastest  
☒ Show advanced devices  
☐ HardCopy compatible only

Available devices:

Name	Core v...	LEs	User I/...	Memor...	Embed...	PLL
EP2C5F256C6	1.2V	4608	158	119808	26	2
EP2C8F256C6	1.2V	8256	182	165888	36	2
EP2C15AF256C6	1.2V	14448	152	239616	52	4
EP2C20F256C6	1.2V	18752	152	239616	52	4

Companion device:  
HardCopy:   
☒ Limit DSP & RAM to HardCopy device resources

< Back Next > Finish Cancel

Рис. 1.6. Вибір типу мікросхеми

**2.6.** Четверте вікно майстра дозволяє указати САПР Quartus II додатково використовувані САПР сторонніх виробників під час створення даного проекту (рис. 1.7). Під час виконання лабораторних робіт використовується тільки САПР Quartus II, тому пропустить цей крок, натиснувши кнопку **Next** для продовження.

Specify the other EDA tools -- in addition to the Quartus II software -- used with the project.

Design Entry/Synthesis  
Tool name: <None>  
Format:   
☐ Run this tool automatically to synthesize the current design

Simulation  
Tool name: <None>  
Format:   
☐ Run gate-level simulation automatically after compilation

Timing Analysis  
Tool name: <None>  
Format:   
☐ Run this tool automatically after compilation

< Back Next > Finish Cancel



Рис. 1.7. Налаштування четвертої сторінки майстра **New Project Wizard**

**2.7.** П'ята сторінка містить повну інформацію про зроблені призначення. При необхідності внесення виправлень, можна повернутися до попередніх сторінок, натиснувши кнопку **Back**.

На рис 1.8. представлена підсумкова сторінка. Натисніть кнопку **Finish**. Проект створений.

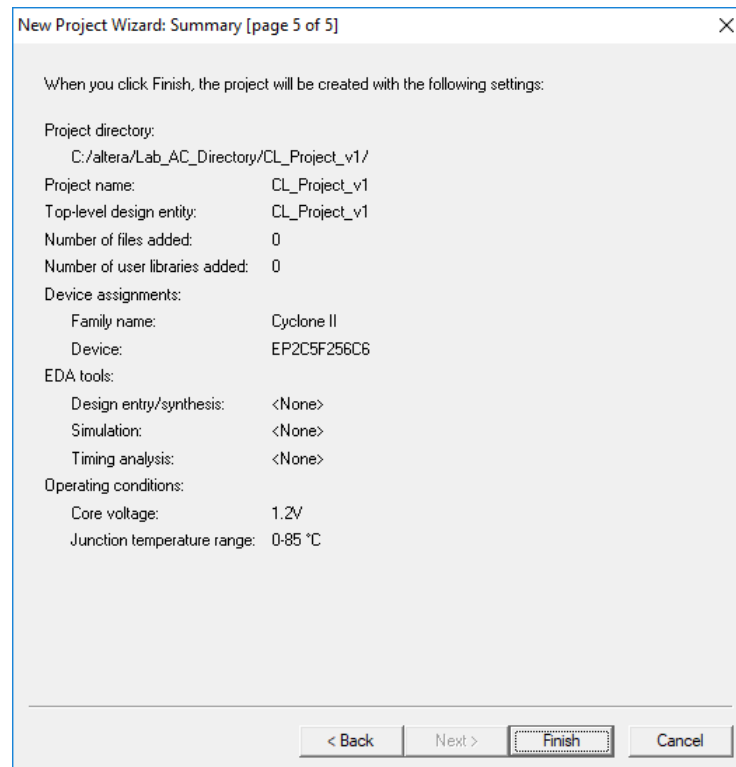


Рис. 1.8. Підсумкова сторінка роботи майстра створення проектів.

**2.8.** Для продовження виконання лабораторної роботи не потрібно виходити із середовища **Quartus II**. Після створення проекту в директорії проекту автоматично генерується ряд файлів, серед яких файл проекту (**Project File**) з розширенням **\*.qpf**. Для лабораторної роботи, що виконується, це файл **CL\_Project\_v1.qpf**. Закритий проект завжди можна відкрити за допомогою команди **Open Project** (меню **File**). У вікні провідника слід обрати каталог проекту і головний файл проекту (рис. 1.9).

**Тут і далі: корисні поради**

Майстер **New Project Wizard** автоматично генерує наступні файли з установками проекту:

- Файл проекту (**Project File**) за замовчанням отримує ім'я **<ім'я проекту>.qpf**. Файл проекту зберігає конфігурацію проекту для компілятора. Під час проектування цей файл можна редагувати в текстовому редакторі.
- Файл з призначеннями проекту **<ім'я проекту>.qdf (Default File)**, — файл генерується і використовується за замовчанням якщо користувач не робить ніяких призначень під час створення проекту. Призначення, які зроблені користувачем зберігаються в файлі з призначеннями **<ім'я проекту>.qsf**



| (*Setting File*), призначення можна редагувати прямо в файлі.

**2.9.** Після закінчення роботи з майстром **New Project Wizard** в основному вікні програми **Project Navigator** на вкладці **Hierarchy** з'явиться імена використовуваного сімейства ПЛС і назва головного файлу проекту. Одночасно у верхній частині головного вікна програми з'являться шлях до проекту, назва проекту і файлу проекту (рис. 1.10)

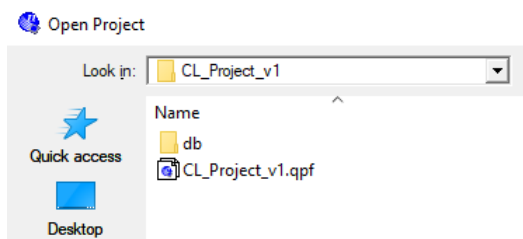


Рис. 1.9. Відкривання існуючого проекту

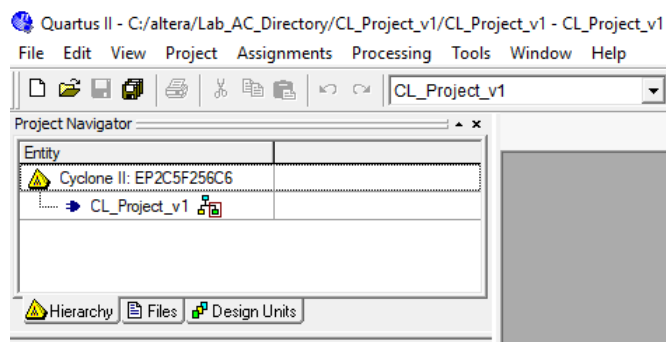


Рис. 1.10. Навігаційна панель після створення нового проекту

### Розроблення функціональних блоків комбінаційних схем

Синтезувати комбінаційні схеми за варіантами завдання. Етапи синтезу комбінаційних схем розглянуті в теоретичній частині до лабораторної роботи.

**3.** Створіть файл верхнього рівня опису проекту. Для чого виконайте наступну послідовність дій:

*Файли опису проекту (Design File)* описують логіку проекту і поєднані між собою ієрархічними зв'язками. Файли опису проекту створюються користувачем в графічному або текстовому редакторі. Файли створені в графічному редакторі зберігаються користувачем в директорії проекту з розширення **\*.bdf**. Завжди існує файл верхнього рівня ієрархії опису проекту, який не слід путати з головним файлом проекту. Назву файлу верхнього рівня ієрархії опису проекту задають під час процедури створення проекту, але сам файл створюється користувачем на початку проектування, після чого йому надається вже задане ім'я. Файл проекту і файли з налаштуваннями проекту не можуть бути файлами верхнього рівня опису ієрархії проекту.

Якщо виникає необхідність інший файл призначити головним файлом опису проекту, слід в навігаційній панелі натиснути правою кнопкою миші на назві файлу і вибрати команду **Set as Top-Level Entity File**. Таким чином інший користувацький файл буде призначений головним файлом опису проекту.

Файл верхнього рівня опису проекту містить блок-схему верхнього рівня проекту – це блок, або декілька блоків, виводи яких приєднуються до виводів мікросхеми. Кожен блок може містити певну кількість вкладених блоків, які в свою чергу можуть мати ще вкладені блоки. Для опису вкладених блоків створюють файли опису проекту наступних рівнів ієрархії. Такий принцип

опису проекту називається *структурним описом*.

**3.1.** В даному циклі лабораторних робіт для вводу проекту в САПР використовуємо графічний редактор. Якщо на навігаційній панелі безпосередньо після створення проекту вибрати вкладку **Files**, ми побачимо, що у складі проекту немає жодного файлу опису проекту (рис. 1.11, а).

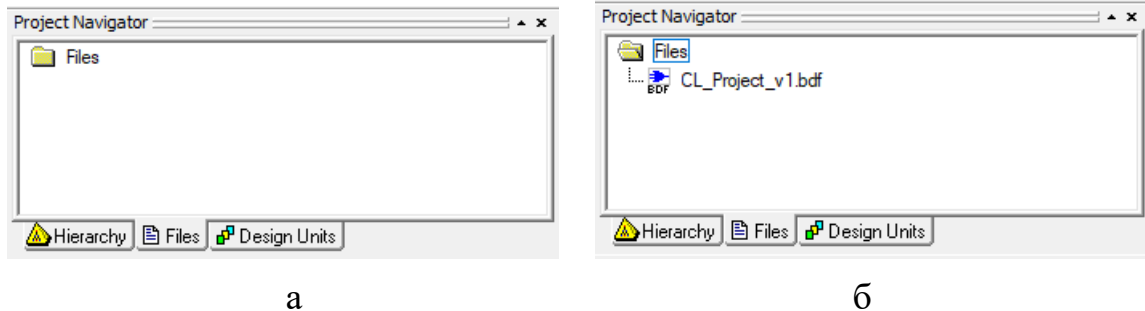


Рис. 1.11. Вкладка **Files**: а - після створення проекту; б – після створення файлу опису верхнього рівня ієрархії проекту

Для створення файлу опису проекту верхнього рівня (оберіть в меню **File** команду **New**, відкриється вкладка **New**, де виберіть тип створюваного файлу (рис. 1.12). Необхідно вибрати вкладку **Design Files** (Файли опису проекту) і у списку, що випадає, обрати рядок **Block Diagram/Schematic File** – графічний файл. Після натискання кнопки **OK** відкриється вікно графічного редактора в головному вікні програми (рис. 1.13). В робочому полі графічного редактора вводять блок-схеми пристроїв та вузлів проекту. При створенні нового файлу проекту в графічному редакторі цей файл отримує ім'я за замовчанням **Block.bdf** (рис. 1.13). Збережіть створений файл опису проекту в директорії проекту з ім'ям файлу проекту – **CL\_Project\_v1.bdf**. Під час першого збереження файлу опису САПР запропонує обрати саме це ім'я, як ім'я файлу верхнього рівня ієрархії опису проекту.

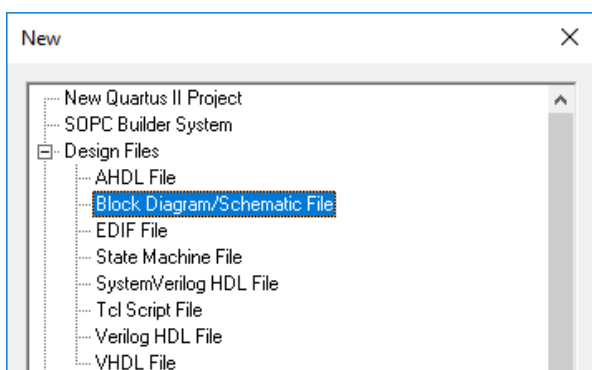


Рис. 1.12. Вікно вибору типу нового файлу проекту

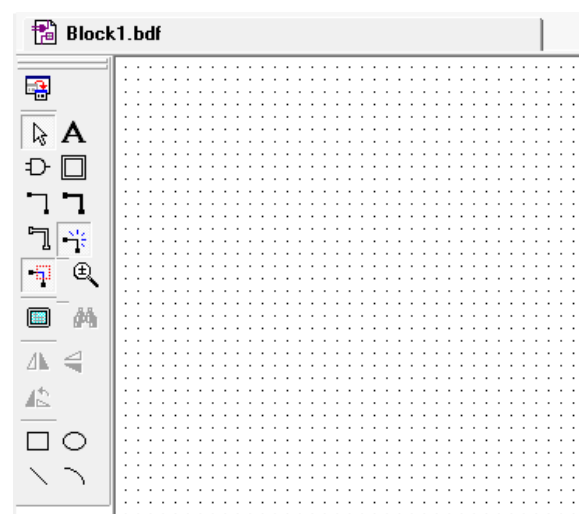


Рис. 1.13. Фрагмент вікна графічного редактора

Тепер на вкладці **Files** навігаційної панелі (рис. 1.9, б) можна побачити ім'я файлу верхнього рівня опису проекту. Можна закрити вікно графічного редактора і відкрити створений файл, натиснувши на імені файлу в навігаційній панелі.

У описаний вище спосіб (команда New в меню File) створюють будь які файли опису проекту. Ієрархія файлів опису проекту відображується на вкладці Files навігаційної панелі вікна САПР (рис. 1.11, б).

Якщо непотрібно виконувати ніяких додаткових дій по обробці проекту, наприклад, компіляцію, можна використовувати команду Open із меню File, яка дозволяє відкрити окремий файл замість всього проекту цілком, або натиснути на імені файлу в навігаційній панелі.

**4.** В робочому полі головного вікна введіть блок схему проекту, для цього використайте панель інструментів, призначення яких наведено на рис. 1.14.


















Назва інструменту	Назва інструменту
 – від'єднання робочого вікна;	 – масштабування об'єктів;
 – виділення;	 – відображення робочого вікна на весь екран;
 – введення тексту;	 – пошук об'єкту;
 – введення стандартних компонент із бібліотеки	 – коригування відображення об'єкту;
 – створення блоку;	 – малювання прямокутника;
 – створення провідників;	 – малювання еліпса;
 – створення шин;	 – малювання прямих ліній;
 – створення каналів зв'язку;	 – малювання дуг.
 – автоматичне під'єднання провідників;	

Рис. 1.14. Призначення піктограм панелі інструментів графічного редактора

**4.1** Виберіть піктограму «Створення блоку» (**Block Tool**) для створення робочого блоку функціонального елементу – для даної лабораторної роботи це функціональний блок (ФБ), що реалізує певну логічну функцію. Натиснути на білій частині поля проекту ліву кнопку миші, і накреслити прямокутну область. Можна використовувати команди відміни (**Undo**) і повторення (**Redo**) дій із меню **Edit**. Вигляд створеного блоку показаний на рис. 1.15.

**4.2.** Кожному блоку та елементу на схемі надайте ім'я. Виберіть піктограму «Виділення» (**Selection Tool**) і клацніть правою кнопкою миші на створеному блоці. Відкриється діалогове вікно «Властивості блоку» (**Block Properties**), де на вкладці **General** в полі **Name** введіть ім'я ФБ, наприклад, *LC1* (*Logic Circuit*) (рис. 1.16). Поле **Instance name** можна залишити без змін, це внутрішній ідентифікатор елементу в проекті.

На вкладці **I/Os** призначте вхідні і вихідні сигнали блоку *LC1* (рис. 1.17) – в

полі **Name** задайте ім'я сигналу, а в полі **Type** його тип: вхідний (**INPUT**), вихідний (**OUTPUT**) або двоспрямований (**BIDIR**). Після заповнення текстових полів натисніть кнопку **Add** і перейдіть до внесення наступного сигналу. Після внесення всіх сигналів натисніть кнопку **OK**. Налаштований блок зображений на рис. 1.18.

Для позначення провідника в полі **Name** вікна **Block Properties** достатньо вказати лише ім'я сигналу, для позначення шини в квадратних дужках слід вказати її розрядність, наприклад **Data[3..0]**.

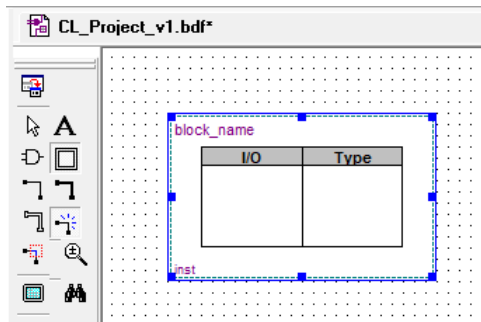


Рис. 1.15. Створення нового блоку

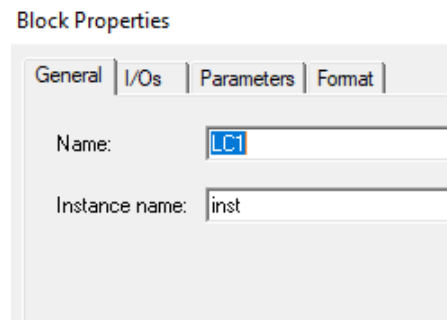


Рис. 1.16. Призначення ім'я новому блоку

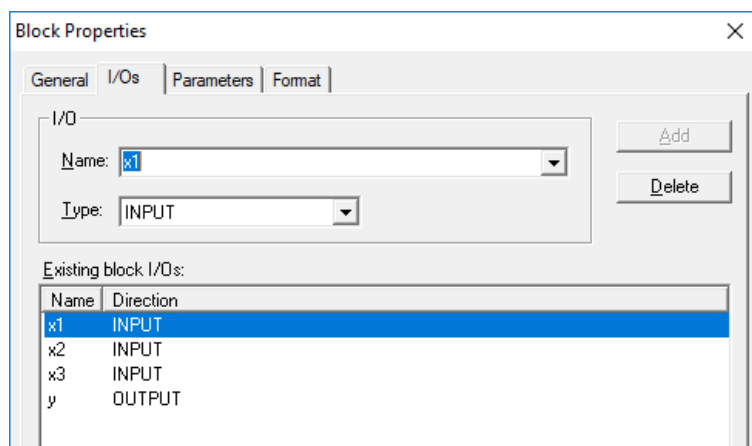


Рис. 1.17. Призначення сигналів введення/виведення

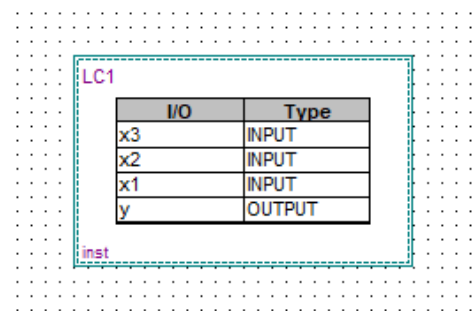


Рис. 1.18. Налаштований блок

#### 4.3. Створіть вхідні та вихідні контакти - піни (**Pins**) функціонального блоку.

*Примітив Quartus* це найдрібніша одиниця схеми, яка реалізована апаратно на кристалі FPGA. Будь яка схема, що введена користувачем в САПР, спеціальними засобами САПР буде перетворена у примітиви, які в подальшому будуть розміщені в комірки мікросхеми.

Примітиви реалізовані в Quartus II, як бібліотечні компоненти, які знаходяться в бібліотеці стандартних компонентів. Для визивання бібліотеки стандартних компонентів натисніть піктограму «Інструмент для створення символу», після чого відкриється вікно «Символ» (**Symbol**), зовнішній вигляд, якого зображений на рис. 1.19.

Контакти введення/виведення та логічні елементи реалізовані як примітиви *Quartus II*.

Контакти введення/виведення (**Pins**) є бібліотечними компонентами, які знаходяться в бібліотеці стандартних компонентів *Quartus II* в розділі *../libraries/primitives/pin* (рис. 1.19).

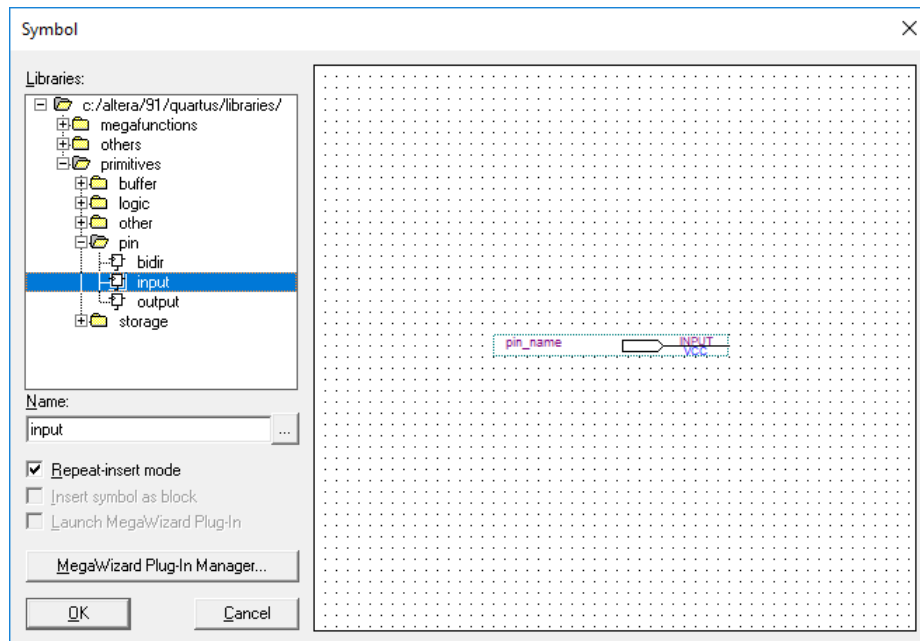


Рис. 1.19. Вікно інструменту для додавання бібліотечних символів функціональних елементів

Виберіть необхідний елемент із списку (для виконання завдання необхідно вхідні контакти (**Input**) для вводу аргументів і вихідні (**Output**) для виводу результату). Зображення вибраного елементу відобразиться в головному полі вікна **Symbol** (рис. 1.19). Натисніть кнопку ОК. Після натискання кнопки ОК вікно **Symbol** закривається, за натисканням у робочому полі редактора вибраний елемент відобразиться в визначеному місці. Встановлення прапорцю **Repeat-Insert Mode** дозволяє розмістити в робочому полі редактора декілька вибраних бібліотечних компонентів. Для відкріплення компонента від курсора миші натисніть правою кнопкою миші на робочому полі і виберіть команду **Cancel** в контекстному меню.

Розмістіть необхідну кількість контактів на робочому полі. Додайте три вхідних контакти (**Input**) для вводу аргументів і один вихідний (**Output**) для виводу результату виконання логічної функції. Результат додавання контактів зображений на рис. 1.20.

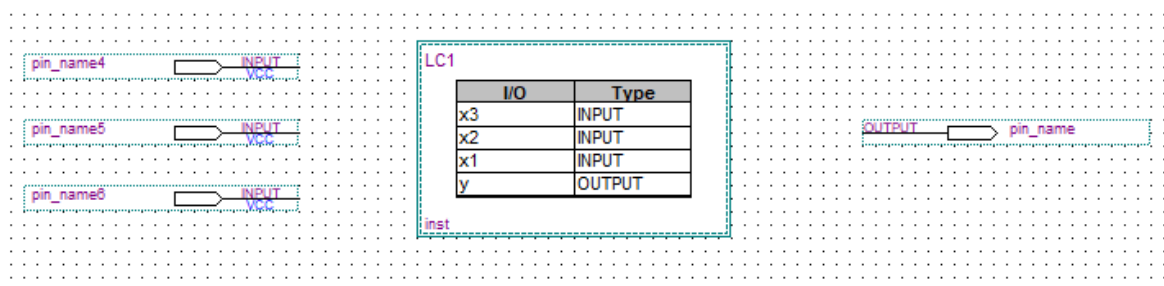


Рис. 1.20. Додавання контактів введення/виведення

4.4. Призначте ім'я кожному контакту. Для цього натисніть правою кнопкою миші на одному з контактів. Оберіть команду **Properties** із контекстного меню. У вікні, що відкрилося, на вкладці **General** введіть ім'я контакту в текстовому полі **Name** (рис. 1.21).

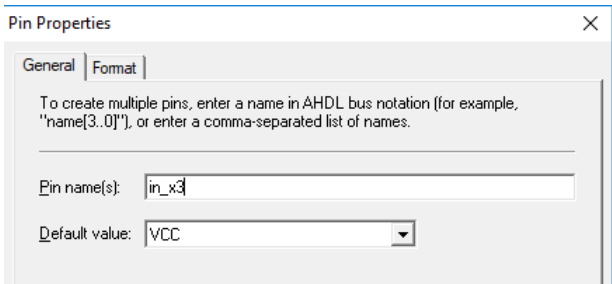


Рис. 1.21. Інструмент для призначення імені контакту

В полі **Default value** задається початкове значення сигналу: логічна одиниця **VCC** (встановлено за замовченням) або логічний нуль (**GND**). У випадку багаторозрядних контактів слід вводити ім'я контакту в форматі шини – із вказанням розрядності контакту в квадратних дужках, наприклад, **name[3..0]** або безпосередньо список всіх контактів.

Призначте ім'я всім контактам функціонального блоку згідно табл. 1.3. Результат перейменування контактів представлений на рис. 1.22.

Таблиця 1.3.

Приклад призначення імен контактам блоку суматора

Ім'я контакту (Name)	Тип контакту (Type)	Коментар
in_x3	INPUT	Аргументи логічної функції
in_x2	INPUT	
in_x1	INPUT	
out_y	OUTPUT	Результат логічної функції

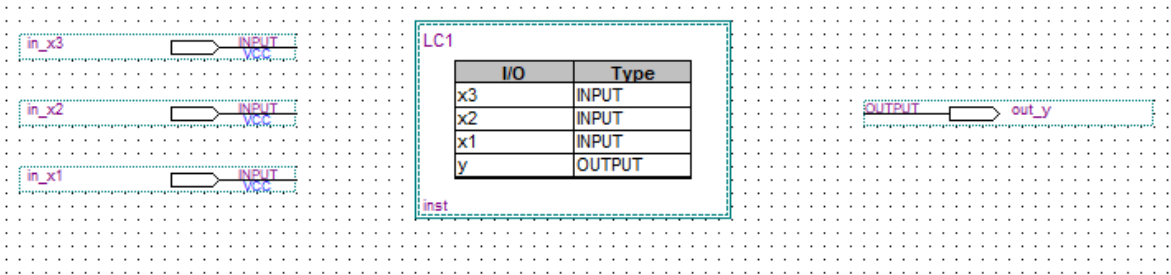


Рис. 1.22. Призначення імені контактам введення/виведення

4.5. Додайте зв'язки між функціональним блоком і контактами. Для створення провідників натисніть на піктограму визивання інструмента для створення провідників (**Node Tool**). Для під'єднання вхідних контактів до входів функціонального блоку використайте інструмент **Conduit** (Канал зв'язку), або просто під'єднайте вхідний контакт до входу блоку, після чого відтягніть його – канал зв'язку створиться автоматично. Результат з'єднання зображено на рис. 1.23.

Всім провідникам і шинам (**Node і Bus**) задайте ім'я. Для цього відкрийте вікно **Properties**, із контекстного меню кожної шини. Рекомендовано, щоб назва провідника/шини співпадала з назвою сигналу на виводі блоку (необов'язково).

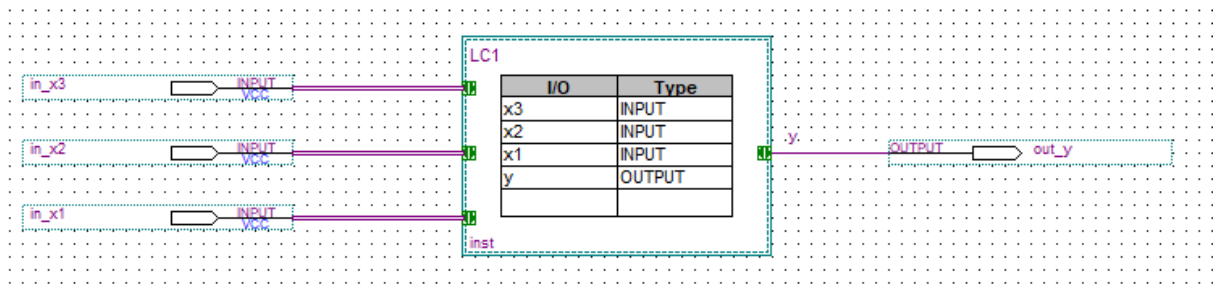


Рис. 1.23. Приклад створення з'єднань між функціональними елементами.

Для створення шин використайте піктограму інструменту для створення шин (**Bus Tool**). Всім провідникам і шинам (**Node і Bus**) необхідно задати ім'я.

Для створення з'єднань в явному вигляді використовуються карта з'єднань (*mappers*), при цьому імена виводів блоків ставляться у відповідність до імені сигналів на шині і провідниках.

САПР *Quartus II* припускає автоматичне з'єднання з використанням режиму *Smart* – якщо імена сигналів входів/виходів одного блоку збігаються з іменами відповідних сигналів іншого блоку, підключення виконується автоматично.

Створіть карту з'єднань. Для створення карти з'єднань натисніть правою кнопкою миші на одному з зелених прямокутників і місці приєднання провідника до функціонального блоку. Відкривається вікно «Властивості з'єднання» (**Mapper Properties**) (рис. 1.24).

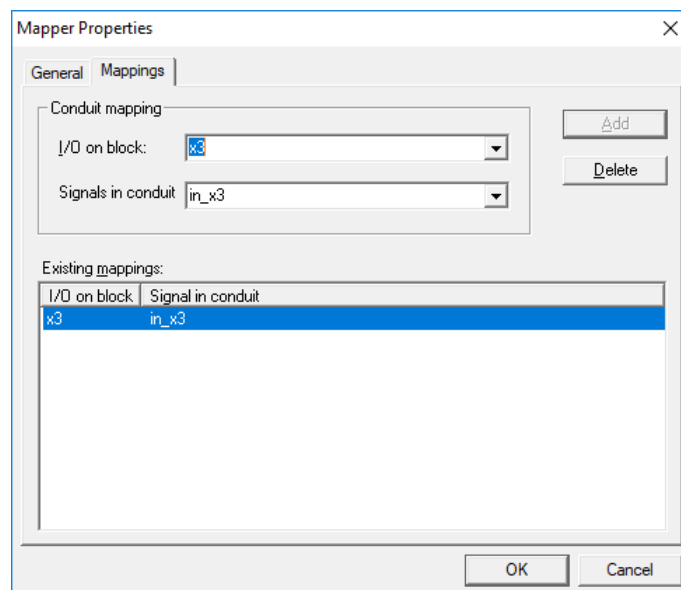


Рис.1.24. Інструмент для створення карти з'єднань

На вкладці **General** виберіть тип виводу блоку із списку **Type** (рис. 1.24). На вкладці **Mapping** задайте карту з'єднань. Для цього в випадаючому списку «Входи



і виходи блоку» (**I/O on block**) виберіть ім'я відповідного виводу блоку. В випадаючому списку «Сигнали в каналі» (**Signals in conduit**) виберіть ім'я сигналу в лінії зв'язку. Натисніть кнопку **Add**, після чого відповідний запис з'явиться в полі «Наявні з'єднання» (**Existing mappings**) (рис. 1.24). Задайте карти з'єднань для всіх інших виводів функціонального блоку. Результат зображений на рис. 1.25.

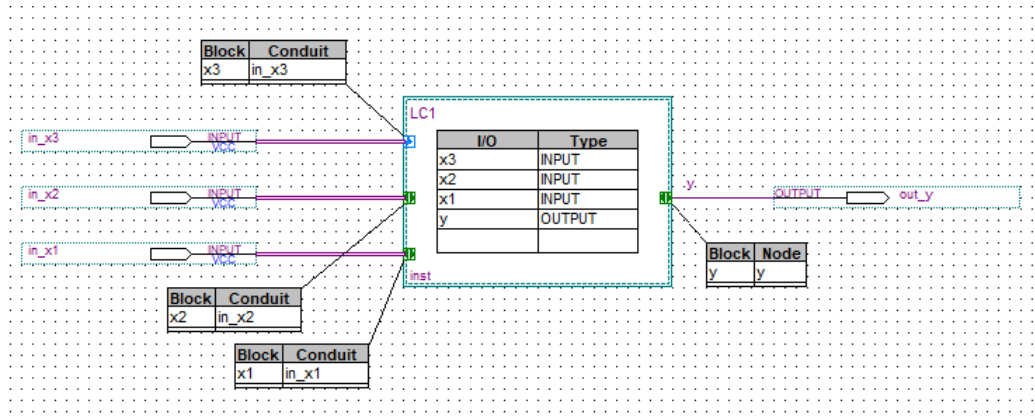


Рис. 1.25. Карти з'єднань

## 5. Створіть файл опису наступного півня ієрархії

**5.1.** Створіть файл опису роботи функціонального блоку **LC1**. Для цього натисніть правою клавішею мишки на блоці **LC1** та в контекстному меню виберіть команду «**Create Design File from Selected Block...**», відкриється вікно створення нового файлу опису виділеного блоку. В списку **File type** виберіть спосіб створення нового файлу, в нашому прикладі, це **Schematic** для створення файлу опису в графічному редакторі (рис. 1.26). Інші пункти списку призначені для створення файлів опису на відповідних мовах програмування апаратури. Зверніть увагу, що САПР пропонує назву файлу опису відповідно до назви блоку, що описується. Можна задати власне ім'я файлу. Далі відкриється нова вкладка графічного редактора з заданим ім'ям (рис. 1.27). Новий файл з'явиться в навігаційній панелі (рис. 1.28).

Зверніть увагу, що знову створений файл опису містить контакти для приєднання до файлу верхнього рівня ієрархії, які були створені під час створення блоку верхнього рівня (рис. 1.27).

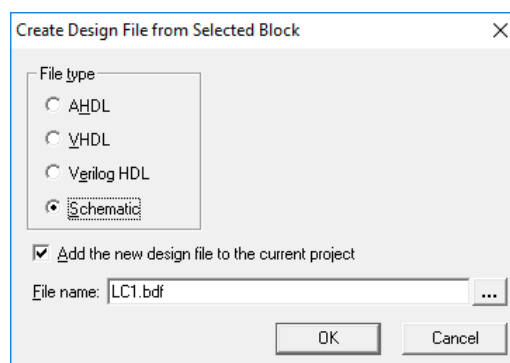


Рис.1.26.

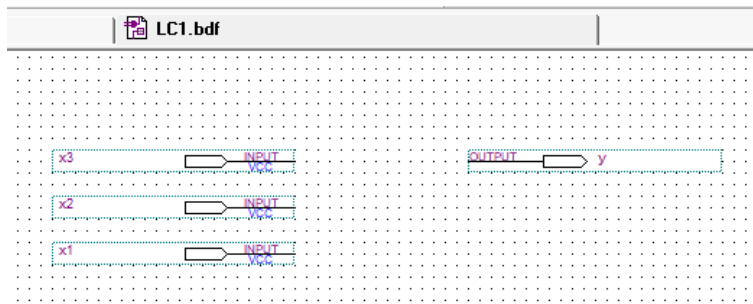


Рис. 1.27.

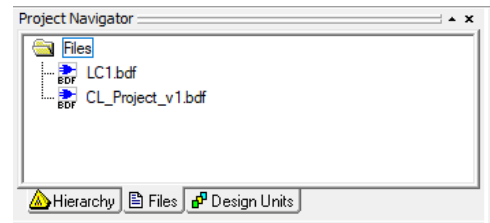


Рис. 1.28.

**5.2.** Створіть логічну (комбінаційну) схему ДДНФ перемикальної функції  $f_4$ , що розроблена під час виконання теоретичного завдання. Для цього додайте на робоче поле файлу **LC1.bdf** елементи логічної схеми – логічні елементи І, АБО, та інші. Приклад комбінаційної схеми в елементному базисі 3І / 3АБО наведено на рис. 1.29, а.

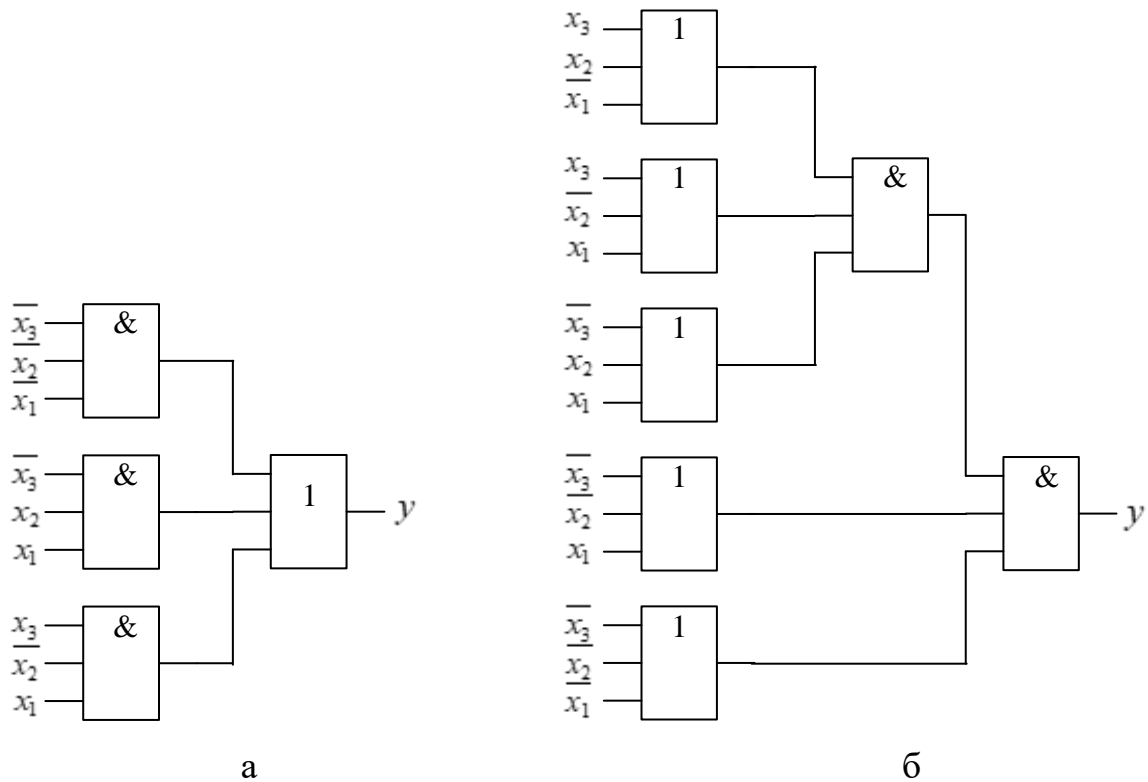


Рис. 1.29. Вихідні комбінаційні схеми: а – ДДНФ перемикальної функції; б – ДКНФ перемикальної функції.

Логічні елементи є бібліотечними компонентами, які знаходяться в бібліотеці стандартних компонентів Quartus II. Для відкривання бібліотеки стандартних компонентів використайте вікно **Symbol**. Логічні елементи є примітивами *Quartus*. Логічні елементи знаходяться в розділі *./libraries/primitives/logic*. Виберіть необхідні елементи із списку. Для виконання завдання виберіть логічні елементи І та АБО з трьома входами – **3and** та **3or** відповідно. Зображення вибраного елемента відобразиться в головному полі вікна **Symbol**. Після натискання кнопки **OK** вікно **Symbol** закривається, за натисканням у робочому полі редактора вибраний елемент відобразиться в

визначеному місці. Розмістіть на робочому полі три елемента **3and** і один елемент **3or**. Аналогічним чином розмістіть на робочому полі п'ять інверторів – елемент **not** також знаходиться в бібліотеці примітивів в списку *../libraries/primitives/logic*.

За допомогою інструмента (**Node Tool**) створити провідники між логічними елементами і виводами блоку. В результаті, на робочому полі створена логічна схема, яка зображена на рис. 1.30.

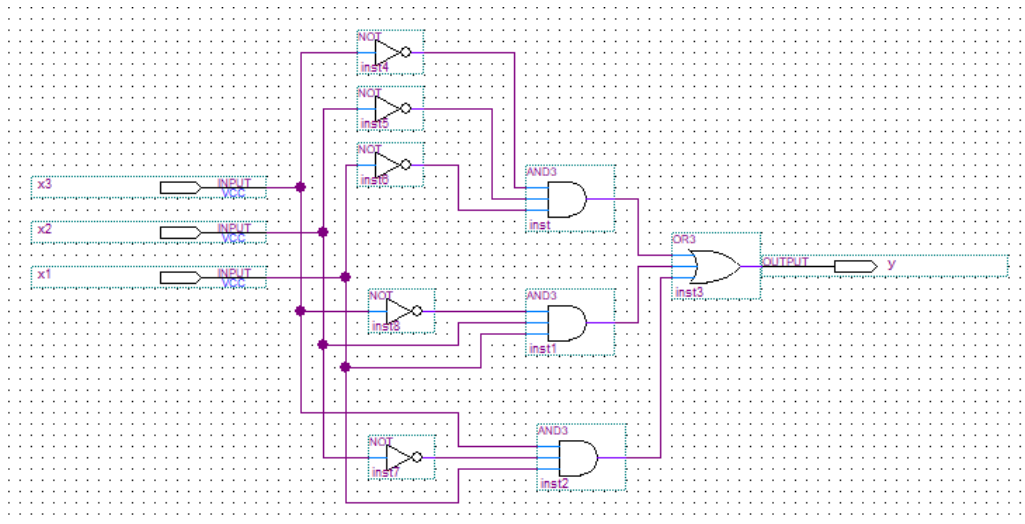


Рис. 1.30. Логічна схема перемикальної функції

Збережіть створену схему. Створений файл можна редагувати, відкривши файл із навігаційної панелі, або із контекстного меню, яке відкривається подвійним натисканням правої кнопки миші на блоці верхнього рівня опису проекту.

Схематично ієрархічний принцип поєднання об'єктів проекту представлений на рис. 1.31.

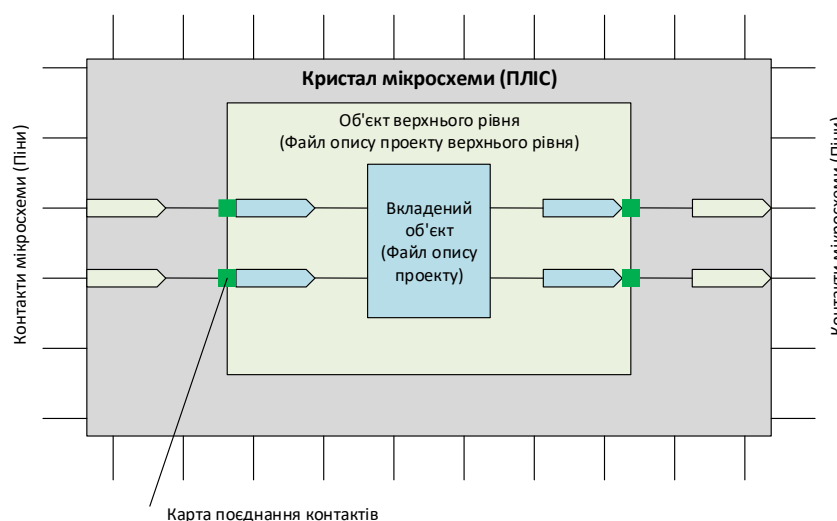


Рис. 1.31. Принцип ієрархічного поєднання об'єктів проекту


## 6. Збережіть і перевірте проект.

Для перевірки коректності створеної схеми виконайте перший етап компіляції **Analysis & Elaboration**. Для цього в меню **Processing** оберіть команду **Start** ⇒ **Start Analysis & Elaboration**.

На першому етапі компіляції **Analysis & Elaboration** виконується коректність зв'язків і підключення пінів у створеній схемі, виконує перевірку наявності всіх файлів в проекті і правильність їх підключень.

Натисніть кнопку ОК у вікні повідомлення, коли аналіз завершиться. Якщо були виявлені будь-які помилки, перевірте всі зв'язки схеми для виправлення помилок.

## 7. Виконайте компіляцію проекту

**7.1.** Якщо перевірка пройшла успішно запустіть повну компіляцію проекту командою **Start Compilation** меню **Processing** або натисніть піктограму , розташовану на панелі інструментів. Після завершення процесу компіляції відкриється діалогове вікно, в якому натисніть кнопку **OK**.

**7.2.** Проаналізуйте інформацію про використані ресурси і часові затримки зі звіту компілятора:

Звіт компілятора надає повну інформацію про результати опрацювання проекту. З його допомогою ви визначає, як компілятор обробляв ваш проект і які отримані результати. Кожна окрема утиліта, що входить до складу компілятора, оновлює інформацію в своїй директорії. Вікно **Compilation Report** відкривається, коли запущено один з додатків, що входять до складу компілятора. По закінченню роботи компілятора відкривається вікно **Flow Summary**.

- З розділу **Flow Summary** звіту компілятора **Compilation Report** (рис. 1.32), випишіть в таблицю (табл. 1.4) наступні значення: **Total logic elements**, **total memory bits**, **total embedded multiplier 9-bit elements** і **total pins**.

Total logic elements	2 / 4,608 (< 1 %)
Total combinational functions	2 / 4,608 (< 1 %)
Dedicated logic registers	0 / 4,608 (0 %)
Total registers	0
Total pins	5 / 158 (3 %)
Total virtual pins	0
Total memory bits	0 / 119,808 (0 %)
Embedded Multiplier 9-bit elements	0 / 26 (0 %)
Total PLLs	0 / 2 (0 %)

Рис. 1.32. Фрагмент розділу **Flow Summary** звіту компілятора **Compilation Report**

Аналіз результатів показує, що для реалізації проекту були задіяні лише логічні ресурси мікросхеми, спеціалізовані ресурси (тобто вбудована пам'ять, вбудований помножувач) не задіяні.

- Розкрийте папку **Fitter** звіту компілятора **Compilation Report**. Виберіть розділ **Resource Section**. Із звіту **Resource Utilization by Entity**, випишіть в таблицю (табл. 1.4) використані ресурси для реалізації функціональних блоків (рис. 1.33).

Fitter Resource Utilization by Entity				
	Compilation Hierarchy Node	Logic Cells	Dedicated Logic Registers	
1	[-] ILab1_SM	2 (1)	0 (0)	(
2	[-] ISM2.inst2	1 (0)	0 (0)	(
3	[-] lpm_add_sub:lpm_add_sub_component	1 (0)	0 (0)	(
4	[-] ladd_sub_03:auto_generated	1 (1)	0 (0)	(

Рис. 1.33. Фрагмент розділу *Resource Utilization by Entity* звіту компілятора *Compilation Reporttpd*

– Оцініть часові затримки за результатами звіту компілятора *Timing Analyzer* із розділів *Timing Analyzer Summary* і *tpd* (рис. 1.34).

Timing Analyzer Summary									
	Type	Slack	Required Time	Actual Time	From	To	From Clock	To Clock	Failed Paths
1	Worst-case tpd	N/A	None	9.813 ns	iny	outs	--	--	0
2	Total number of failed paths								0

tpd					
	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	9.813 ns	iny	outs
2	N/A	None	9.806 ns	iny	outp
3	N/A	None	9.041 ns	inz	outs
4	N/A	None	9.027 ns	inz	outp
5	N/A	None	8.834 ns	inx	outs
6	N/A	None	8.827 ns	inx	outp

Рис. 1.34. Аналіз швидкодії на підставі звіту компілятора *Timing Analyzer*: а – найдовший шлях проходження сигналу; б – всі шляхи проходження сигналів

Розділ звіту компілятора *Timing Analyzer* дозволяє проаналізувати часові затримки проходження сигналів через всі ланцюги схем проекту.

Таблиця 1.4

#### Аналіз звіту компілятора

Звіти компілятора	LC1	LC2
Total logic elements	2 (<1%)	
Total pins	5 (3%)	
LC Block	1 Logic Cells	
Actual Time	9,813 ns	
Найдовший шлях проходження сигналу		
Найкоротший шлях проходження сигналу		
Затримка на логічних елементах	2ns	2ns
Затримка на входних контактах		
Затримки на вихідних контактах		
Кількість задіяних логічних комірок		
Додатково		

## 8. Виконайте симуляцію проекту *CL\_Project\_v1*.

**8.1.** Симуляцію починають зі створення стимуляційного файлу (**Vector Wave Form File**) – *Test Bench* файл, що містить стимули для моделювання. Для цього оберіть

команду **New** меню **File**. В розділі **Verification/Debugging Files** оберіть тип файлу **Vector Wave Form File**. (рис. 1.35). Натисніть кнопку **OK**. В головному вікні з'явиться вкладка з назвою файлу за замовчанням **Waveform.vwf**. На вкладці знаходиться інструмент для створення часової діаграми для моделювання роботи проекту. Збережіть новий файл з ім'ям проекту **CL\_Project\_v1.vwf**.

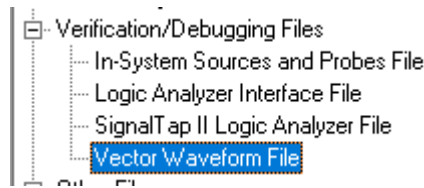


Рис. 1.35. Створення стимуляційного файлу **Vector Wave Form File**

**8.2.** Додайте входні сигнали для моделювання роботи схеми. Для цього в полі **Name** натисніть два рази лівою кнопкою миші. Відкриється вікно **Insert Node or Bus** (рис. 1.36). Якщо назви сигналів відомі введіть їх в полі **Name**.

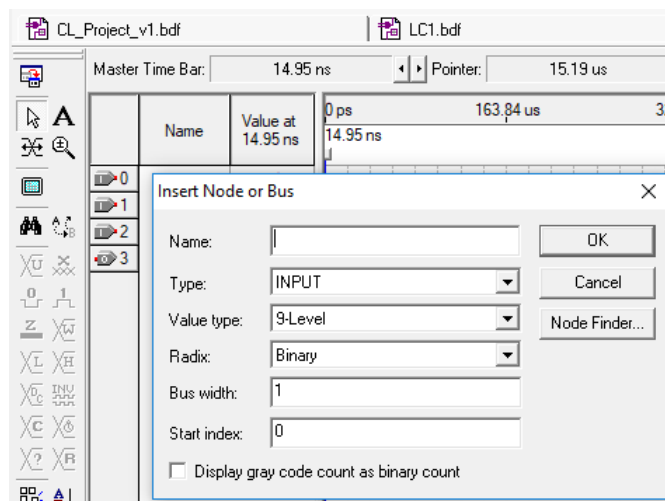


Рис. 1.36. Інструмент **Insert Node or Bus** для додавання сигналів для моделювання.

Інакше скористайтесь інструментом **Node Finder** для пошуку сигналів. Для цього натисніть кнопку **Node Finder** у вікні **Insert Node or Bus**, відкриється відповідне вікно **Node Finder** (рис. 1.37). Скористайтесь фільтром (**Filter**) для вибору групи сигналів (**Pins: input**). Натисніть кнопку **List**. Перелік необхідних входних сигналів відкриється в полі **Nodes Found**. Виберіть необхідні сигнали і перенесіть їх у поле **Selected Nodes** за допомогою кнопки **>**, або всі сигнали за допомогою кнопки **>>**. Натисніть кнопку **OK**.

У вікні **Insert Node or Bus** можна встановити формат входних та вихідних сигналів, для чого із переліку **Radix** виберіть, наприклад **Binary** (двійковий формат), після чого натисніть кнопку **OK**. У робочому полі часової діаграми з'являться входні сигнали для моделювання (рис. 1.38, а). Аналогічно додайте вихідні сигнали (**Pins: output**), після чого в робочому вікні **CL\_Project\_v1.vwf** з'являться і вихідні сигнали (рис. 1.38, б).

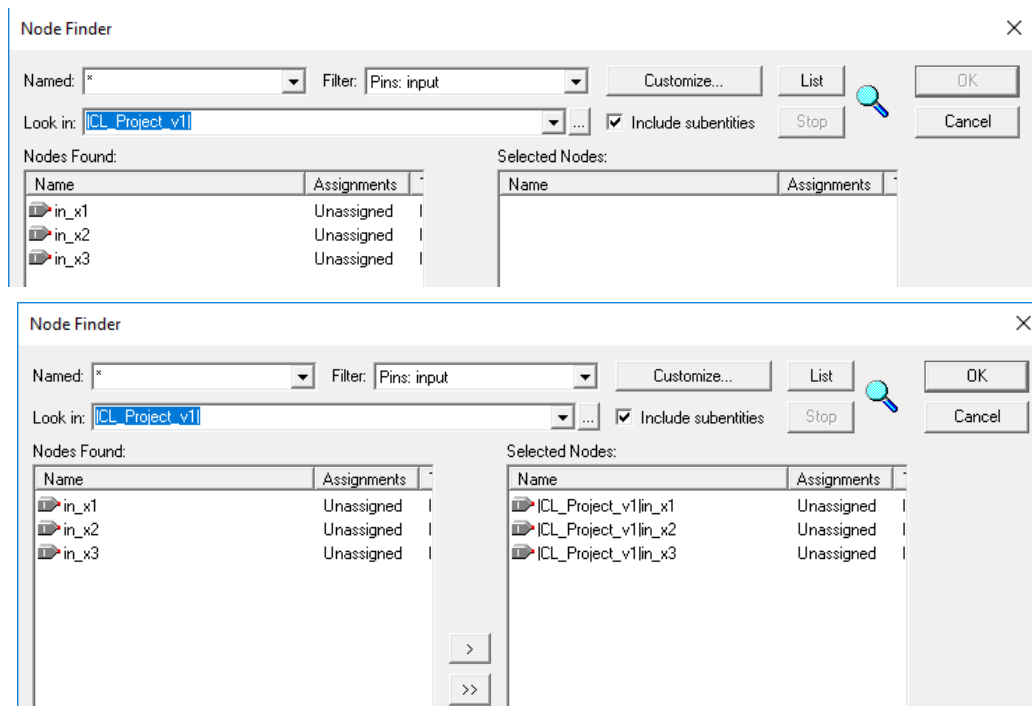
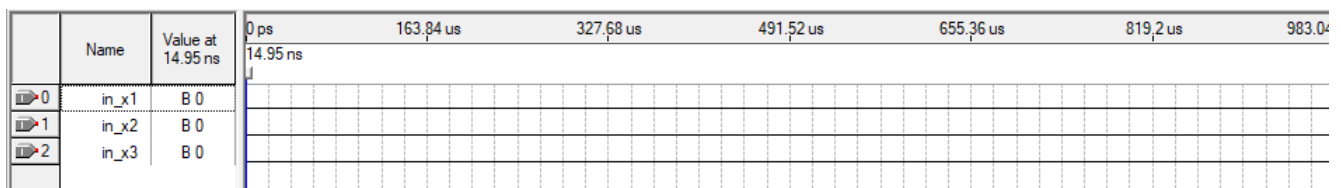
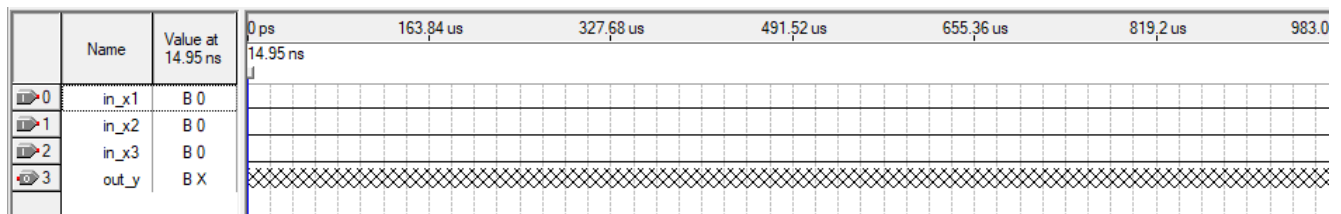


Рис. 1.37. Інструмент **Node Finder** для пошуку сигналів для моделювання



*a*



*б*


Рис. 1.38. Вхідні та вихідні сигнали для моделювання роботи схеми


Сигнали у вікні **Name** можна перетягувати кнопкою миші для розміщення і угруповання для зручності подальшого сприйняття результатів моделювання, наприклад, доцільно розподіляти по окремим групам вхідні, керуючі та вихідні сигнали.




Формат виводу сигналів на діаграмі можна змінити в будь який момент після формування діаграми, а також і під час перегляду результатів моделювання. Для цього слід вибрати необхідний сигнал в полі **Name** і натиснути правою кнопкою миші на назві сигналу або на виділеній ділянці. У вікні **Node Properties**, яке відкриється, із переліку **Radix** слід вибрати необхідний формат сигналу, після чого натиснути кнопку **OK**. Зручними для перегляду результатів





моделювання є формати **Unsigned Decimal** –десяткове число без знаку і **Signed Decimal** – десяткове зі знаком.

**8.3.** Задайте інтервал для моделювання, для цього скористайтесь командою **End Time** із головного меню **Edit**. У вікні, що відкрилося, у полі **Time** задайте значення **1ms** (**s** – секунда, **ms** – мілі, **us** – мікро, **ns** – нано, **ps** – піко). Натисніть кнопку **OK**. Загальний інтервал для моделювання встановиться в **1ms**. За допомогою інструменту  можна зменшувати або збільшувати масштаб часової діаграми: натискання лівої кнопки миші збільшує зображення, правої – зменшує.

**8.4.** Задайте значення вхідних сигналів для моделювання. Для цього оберіть інструмент  і виберіть необхідний сигнал, натиснувши на його імені в полі **Name**. Для завдання значень сигналів використовуйте піктограми на панелі інструментів.

Простим способом для завдання значень сигналів є виділення мишею необхідного часового проміжку і встановлення значення сигналу за допомогою інструментів на панелі інструментів. Якщо треба встановити весь діапазон часу модулювання в визначене значення слід натиснути на імені сигналу в полі **Name**. Для встановлення виділеного діапазону в значення логічної 1 або логічного 0 використовують піктограми  та  відповідно. Для встановлення значень сигналів на визначеному проміжку часу використовують інструмент **Arbitrary Value**, який визивається піктограмою . У вікні, що відкривається, можна задати точний проміжок часу і значення сигналу в необхідному форматі.

На часовій діаграмі для сигналу **in\_x3** встановіть значення логічної 1 на проміжку часу тривалістю від **40ns** до **80 ns**. Для цього скористуйтеся інструментом **Arbitrary Value**. Для цього натисніть піктограму , у вікні, що відкрилося, у полі **Start Time** задайте значення **40ns**, а у полі **End Time** – **80 ns**. У полі **Radix** оберіть значення **Binary**, у полі **Numeric or Name Value** задайте значення логічної 1. Натисніть кнопку **OK**.

Для завдання регулярних послідовностей сигналів використовують інструмент  - **Count Value** на панелі інструментів або в контекстному меню, що випадає під час натискання правої кнопки миші на виділеному полі часової діаграми. Для цього в контекстному меню слід вибрати команду **Count Value** в списку команд **Value**. У вікні, що відкривається на вкладці **Counting** слід задати початкове і наступне значення послідовності, а на вкладці **Timing** слід задати початок періодичної послідовності – **Start Time**, кінець періодичної послідовності – **End Time** і тривалість кожного проміжку переключення сигналу – **Count every**.

За допомогою інструменту **Count Value** на проміжку часу від **0ns** до **80ns** задайте періодичні значення двійкових сигналів від 0 до 1 на кожних **20ns** лінії сигналу **in\_x2** і на кожних **10ns** лінії сигналу **in\_x1**. (рис. 1.39). Отримана часова діаграма зображена на рис. 1.40. Збережіть файл **CL\_Project\_v1.vwf**.

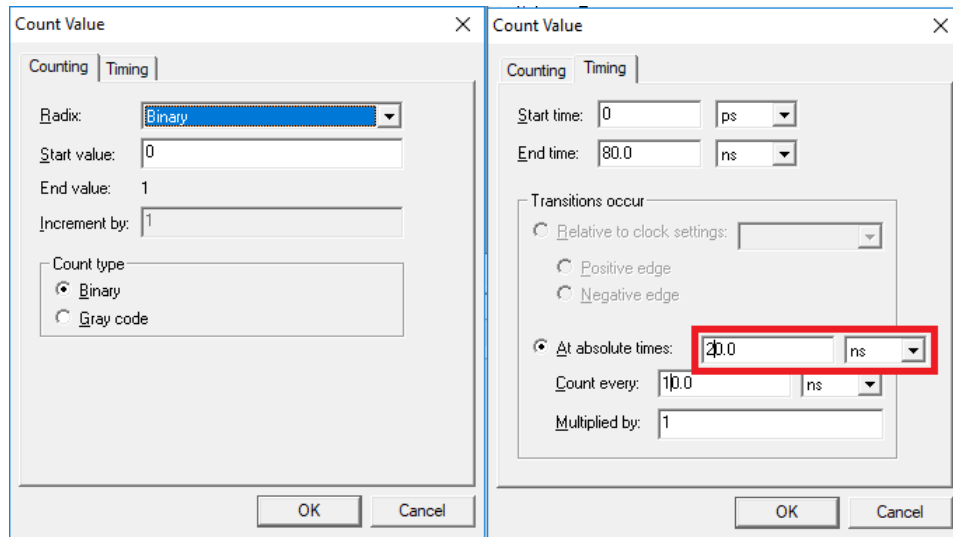


Рис. 1.39. Встановлення періодичних значень вхідних сигналів.

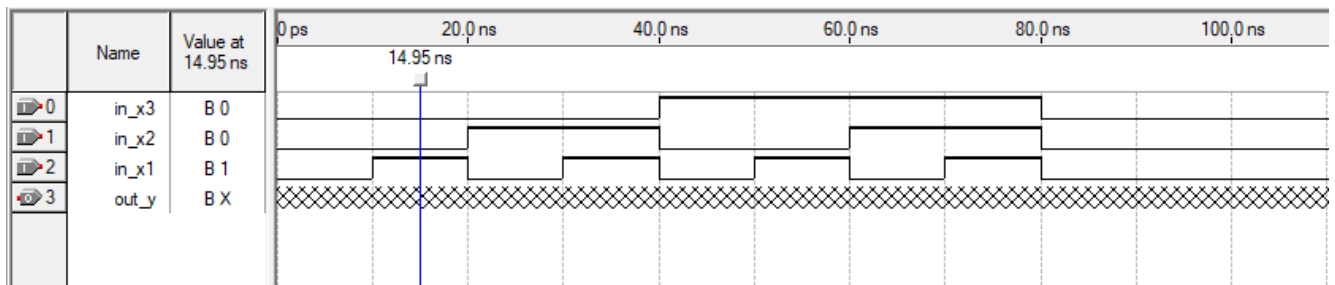



Рис. 1.40. Встановлення вхідних сигналів для моделювання на часовій діаграмі

**8.5.** Для запуску симуляції використайте піктограму  в панелі задач, або виберіть команду **Start Simulation** в меню **Processing**. За успішно виконаного процесу симуляції відкривається відповідне повідомлення і відкривається вікно звіту симулятора **Simulation Report**. Результат виконаної симуляції проекту *Lab2\_SM\_v2* представлений на рис. 1.41.

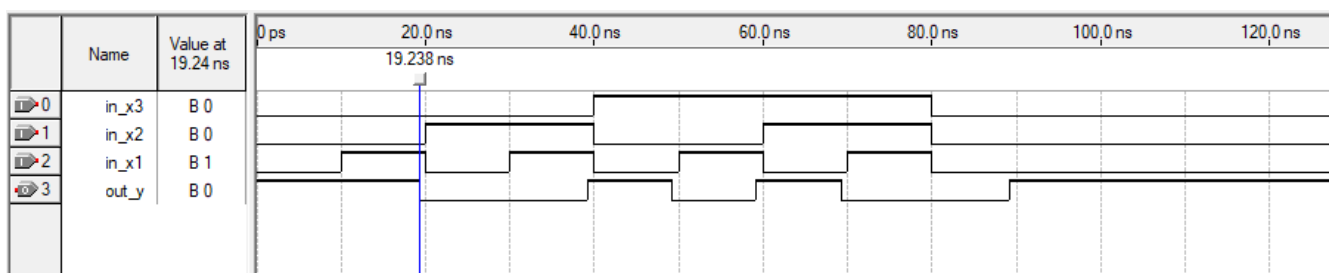
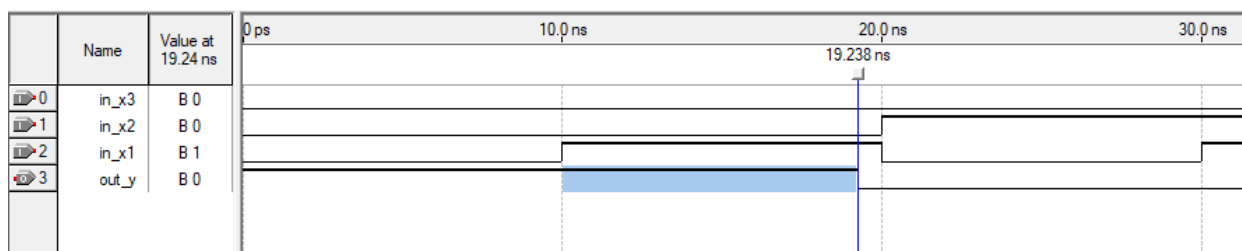



Рис. 1.41. Результати симуляції




1.42. Аналіз затримки формування результату

**8.6.** Проаналізуйте правильність роботи схеми. На діаграмі (рис. 1.41) можна виміряти затримку формування результату – приблизно  $9ns$ , впевнитися, що це співпадає з теоретичними оцінками, які були сформовані під час компіляції (рис. 1.34).

Затримки під час моделювання роботи цифрових пристроїв в САПР *Quartus II* на 90% наближаються до показників реальних пристроїв. Затримки під час моделювання розраховані на підставі усереднених показників реальних пристроїв і використовуються як моделі під час часового моделювання.

Під час моделювання можна відключити врахування затримок формування вихідних сигналів на елементах пристроїв, для цього слід вибрати режим **Functional** у вікні налаштування симуляції, для цього в головному меню слід обрати команду **Assignment** і в меню, що відкрилося, вибрати команду **Setting** (піктограма  на панелі інструментів). У вікні, що відкрилося у списку **Simulation Mode** необхідно вибрати режим **Functional**. Перед виконанням функціональної симуляції необхідно виконати команду **Generate Functional Simulation Netlist** із головного меню **Processing** для автоматичного налаштування симулятора.

Функціональне моделювання відображує правильність роботи схеми і рекомендоване для виконання першого етапу симуляції складних схем. Функціональне моделювання зручно використовувати для пошуку функціональних помилок в роботі пристрою. Після функціонального моделювання виконують етап моделювання з врахуванням часових затримок, для цього у списку **Simulation Mode** необхідно вибрати режим **Timing** (в меню **Assignment** вибрати команду **Setting**). Режим встановлений за замовчанням в налаштуваннях симулятора.

**8.7.** Перейдіть в режим функціонального моделювання **Functional**. Для цього в меню **Assignment** виберіть команду **Setting** і у списку **Simulation Mode** виберіть режим **Functional**. Виконайте команду **Generate Functional Simulation Netlist** із головного меню **Processing**. Для запуску симуляції використайте піктограму  в панелі задач, або виберіть команду **Start Simulation** в меню **Processing**. Результат виконаної функціональної симуляції проекту *CL\_Project\_v1* представлений на рис. 1.43.

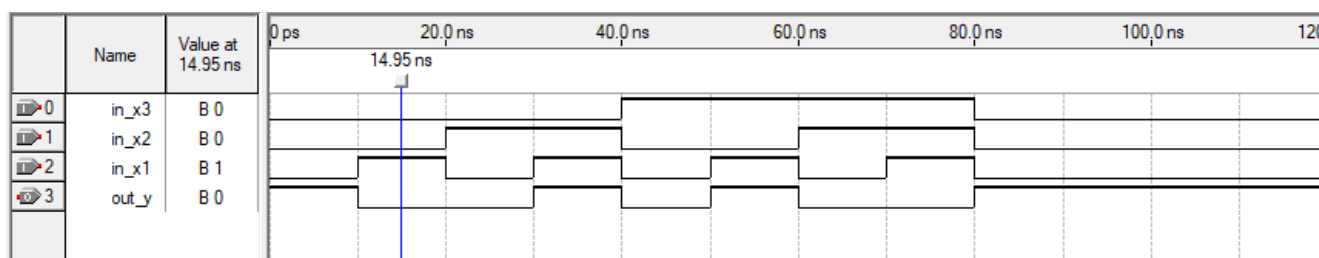


Рис. 1.41. Результати функціональної симуляції

**9.** Виконайте етапи введення, синтезу та симуляції комбінаційної схеми заперечення ДДНФ (або ДКНФ) функції  $f_4$ .

**9.1. Перший спосіб.** Для реалізації ДКНФ функції  $f_4$  можна створити окремий

проект (з назвою **CL\_Project\_v2**) і виконати пункти **1-8** для проектування комбінаційної схеми, приклад якої наведений на рис. 1.29, б.

**9.2. Другий спосіб.** Комбінаційну схему для реалізації ДКНФ функції  $f_4$  (рис. 1.29, б) можна ввести безпосередньо в файлі опису верхнього рівня вже створеного робочого проекту **CL\_Project\_v1**. Виберемо цей спосіб для експериментів в даній лабораторній роботі.

**9.3.** Відкрийте проект **CL\_Project\_v1**, відкрийте файл **CL\_Project\_v1.bdf** (рис. 1.25). Додайте на робоче поле входні та вихідні контакти, наприклад **in2\_x3**, **in2\_x2**, **in2\_x1**, **out2\_y** (рис. 1.42).

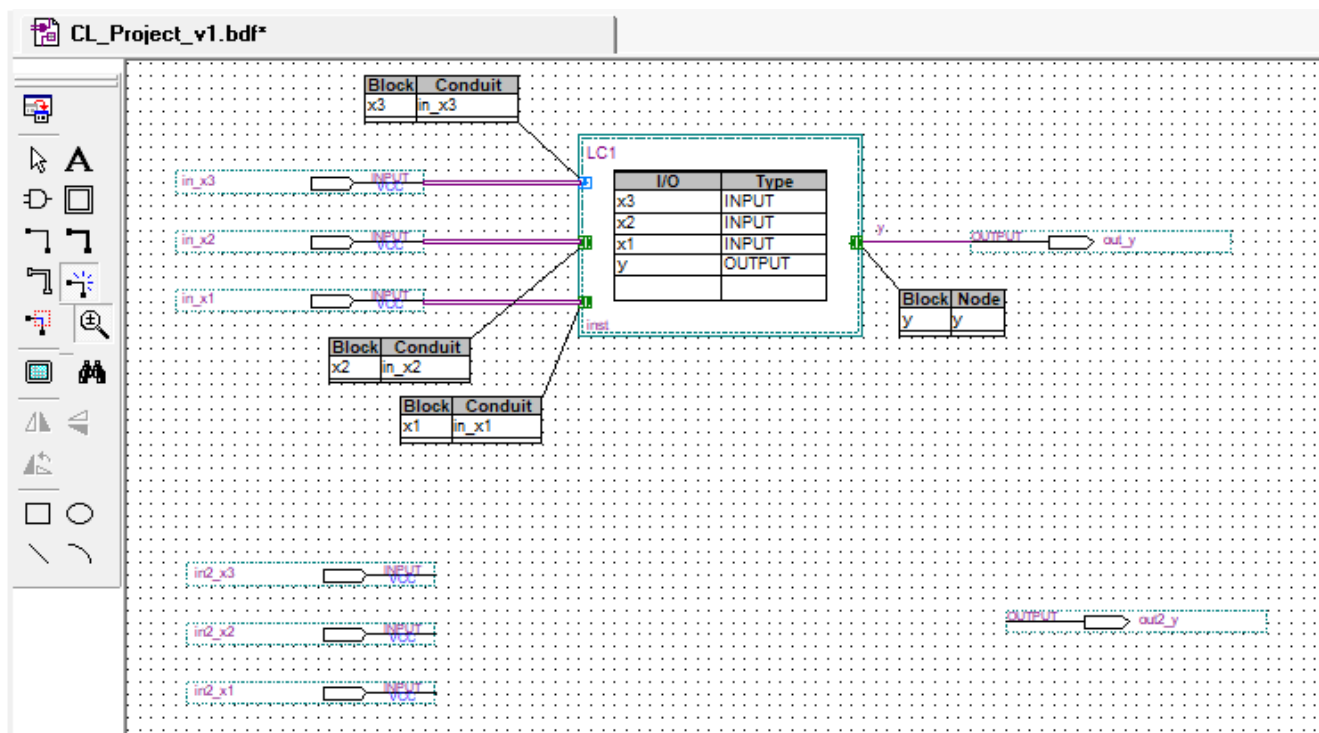


Рис. 1.42. Додавання контактів для логічної схеми ДКНФ перемикальної функції

Якщо Ви вже відчуваєте впевненість у власному професіоналізмі відведіть входні контакти для другої схеми із входних контактів першої комбінаційної схеми. Але вивід другої схеми має бути окремий в будь якому випадку.

В складних схемах припускається розривання провідників та шин. В цьому випадку провідники, які розірвані, повинні мати однакові назви. Особливо це зручно для розгалуження шин, наприклад восьмирозрядна шина **data[7..0]** з одного боку розгалужується на дві чотирирозрядні шини **data[7..4]** і **data[3..0]** (рис. 1.43). Компілятор поєднає провідники відповідно до призначених імен.

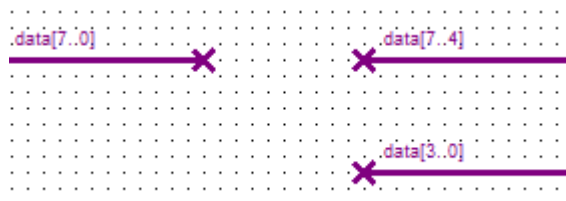


Рис. 1.43.

**9.4.** Введіть комбінаційну схему ДКНФ перемикальної функції  $f_4$  прямо в робочому полі файлу *CL\_Project\_v1.bdf*. Результат представлений на рис. 1.44.

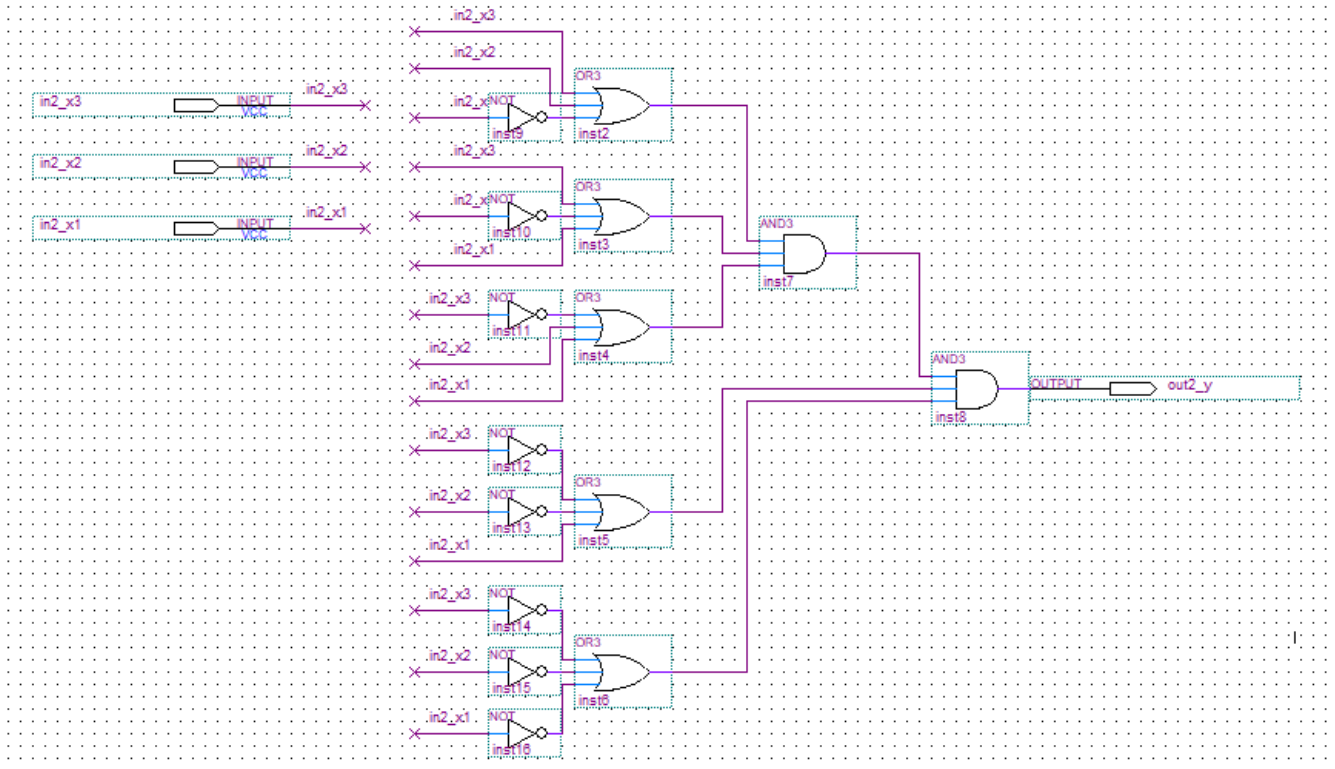


Рис. 1.44. Логічна схема ДКНФ перемикальної функції

**9.5.** Створіть стимуляційний файл (**Vector Wave Form File**) для моделювання. Для цього оберіть команду **New** меню **File**. В розділі **Verification/Debugging Files** оберіть тип файлу **Vector Wave Form File**. (рис. 1.35). Натисніть кнопку **OK**. Збережіть новий файл з ім'ям *CL\_Project\_v2.vwf*.

**9.6.** Додайте вхідні сигнали для моделювання роботи схеми. Для цього у вікні **Insert Node or Bus** (рис. 1.36) оберіть назви сигналів, що належать до схеми ДКНФ перемикальної функції, або оберіть всі сигнали для порівняння роботи двох схем: ДДНФ і ДКНФ перемикальної функції  $f_4$ .

**9.10.** Задайте в налаштуваннях симулятора новий вхідний файл для моделювання, для цього в меню **Assignment** виберіть команду **Setting** і у списку **Simulation Input** задайте ім'я другого файлу зі стимулами для моделювання *CL\_Project\_v2.vwf*.

**9.11.** Виконайте етап функціональної симуляції. Для цього виконайте команду **Setting** головного меню **Assignment**, у вікні налаштування симулятора задайте режим симуляції **Functional**. Виконайте команду **Generate Functional Simulation Netlist** із головного меню **Processing**. Результат виконаної функціональної симуляції проекту представлений на рис. 1.45. На діаграмі представлено результати функціонального моделювання двох схем: ДДНФ і ДКНФ перемикальної функції  $f_4$ . Видно, що результати збігаються, тобто цільова функція реалізована правильно.

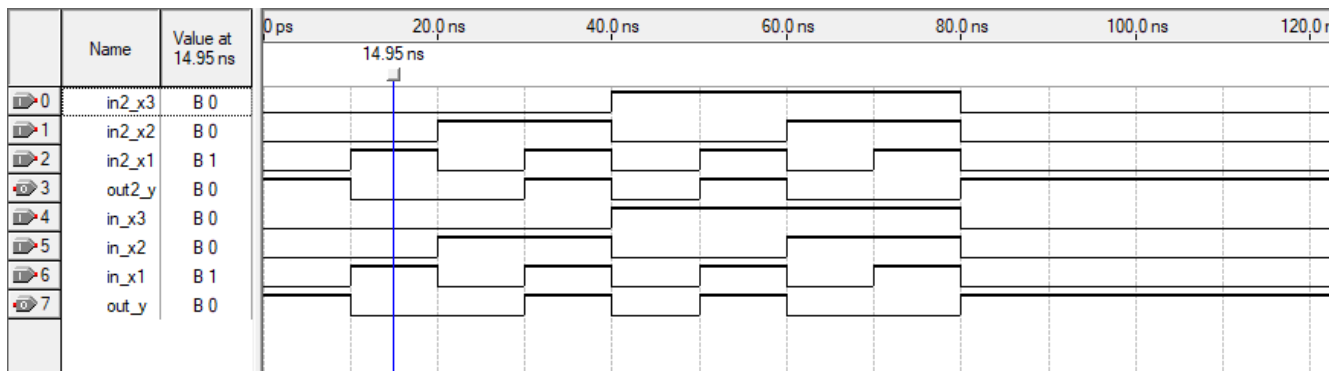


Рис. 1.45. Результати функціонального моделювання схем: ДДНФ і ДКНФ перемикальної функції f4

**9.12.** Дослідить часові параметри роботи розробленої схеми ДКНФ двох схем: ДДНФ і ДКНФ перемикальної функції f4. Для цього виконайте етап часової симуляції з врахуванням затримок. Перейдіть в режим симуляції Timing в налаштуваннях симулятора. Запустіть симуляцію. На діаграмі (рис. 1.45) представлено результати моделювання двох схем: ДДНФ і ДКНФ перемикальної функції f4 з врахуванням часових затримок передавання сигналів. Проаналізуйте результати симуляції, виміряйте час затримки формування результату, порівняйте отримані виміри з результатами компіляції. Результати часового аналізу впишіть в таблицю 1.4. Вкажіть на можливі причини виникнення короточасних помилкових сигналів (просічок). Запропонуйте шляхи боротьби з короточасними помилковими сигналами, виконайте корекцію комбінаційної схеми для подолання просічок.

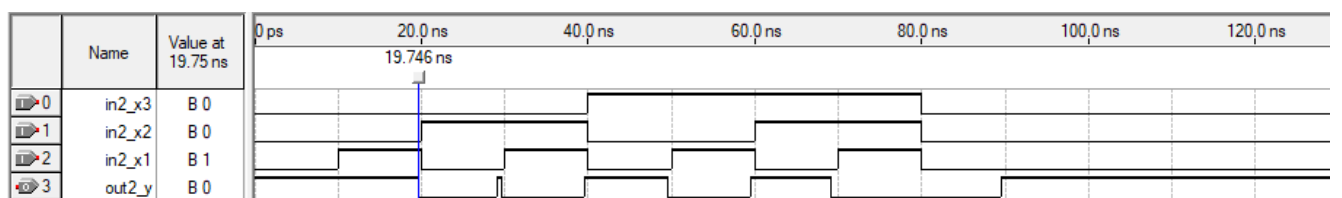


Рис. 1.46. Результати часового моделювання схем: ДДНФ і ДКНФ перемикальної функції f4

Для пошуку помилок часто виникає необхідність в аналізі значень сигналів на проміжних провідниках і шинах. Для визначення значень проміжних сигналів слід підключити до необхідного провідника вихідний контакт і вивести значення сигналу на вивід мікросхеми. Значення сигналу можна переглянути під час моделювання.

Для позбавлення від просічок можна використовувати фільтри, або додаткові елементи (повторювачі).

## 10. Зробіть загальні висновки по роботі