

Лабораторна робота № 3

Компіляція проекту в САПР *Quartus II*. Аналіз результатів компіляції

Ціль роботи:

- Вивчення процесу компіляції проектів в САПР *Quartus II* та отримання навиків аналізу результатів компіляції.
- Створення версій проектів в САПР *Quartus II*, порівняння результатів різних версій проектів.
- Вивчення різних способів введення проектів в САПР *Quartus II*.
- Отримання навиків роботи з редактором призначень.

1. Порядок виконання роботи

1. Створіть нову версію проекту.

Для перевірки впливу різних установок і призначень на кінцевий результат проектування, САПР *Quartus II* дозволяє створювати різні версії проекту зі своїм власним файлом призначень QSF. Це дає можливість створювати й порівнювати між собою результати різних версій проекту.

1.1. Оберіть команду **Revisions** в меню **Project**. У вікні **Revisions**, що відкрилося, натисніть кнопку **Create**. У вікні **Create Revision**, що відкрилося за цим (рис. 2.1), задайте ім'я нової версії проекту. У списку, що випадає, оберіть ім'я батьківського проекту. Всі інші налаштування залиште за замовчанням. Натисніть кнопку **OK**. Вікно **Revision** закриється. У списку, що випадає, на панелі інструментів вікна САПР з'явиться задане ім'я нової версії проекту (рис. 2.2). За допомогою цього списку можна переключатися між створеними версіями проекту. Оберіть проект з назвою **Lab2_SM_v2**.

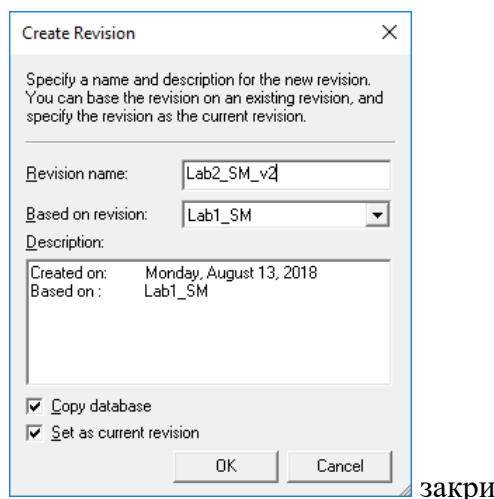


Рис. 2.1. Вікно створення нової версії проекту. Задавання ім'я нової версії проекту

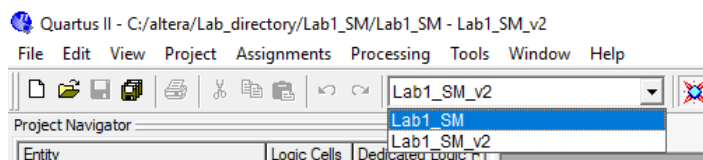


Рис. 2.2. Інструмент для переключення між версіями проекту

2. Реалізуйте повний однорозрядний суматор за допомогою мегафункції.

2.1. Створіть новий графічний файл опису проекту (**Schematic File**) за допомогою команди **New** меню **File**. Збережіть новий файл з назвою **Lab2_SM_v2.bdf** і встановіть його, як файл верхнього рівня ієрархії опису проекту. Це можна зробити в навігаційній панелі, натиснувши на назві файлу правою кнопкою миші. В контекстному меню оберіть команду **Set as top-level entity**. Відкрийте файл верхнього рівня опису проекту **Lab1_SM.bdf**, скопіюйте схему і вставте її в робоче поле файлу **Lab2_SM_v2.bdf**.

2.2. Замість двох напівсуматорів і логічного елементу АБО створіть функціональний блок повного однорозрядного суматора. Для цього видаліть непотрібні блоки і на робочому полі графічного редактора залиште вхідні та вихідні контакти. Скористуйтеся майстром **Mega Wizard plugin manager** в меню **Tools**. Детально процес створення мегафункції описано в пункті 3.10 Лабораторної роботи №1. Для створення повного суматора слід додати вхідний перенос **Carry in** і вихідний перенос **Carry out**. Результат введення функціональної схеми повного однорозрядного суматора наведений на рис. 2.3.

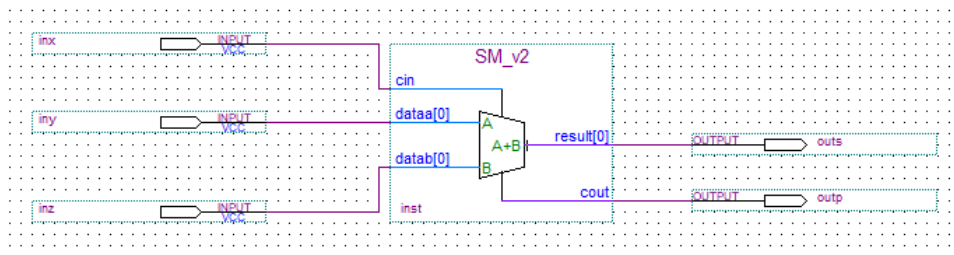



Рис. 2.3. Функціональна схема повного однорозрядного суматора

2.3. Збережіть і перевірте проект. Для перевірки коректності створеної схеми виконайте перший етап компіляції **Analysis & Elaboration**. Для цього в меню **Processing** оберіть команду **Start** \Rightarrow **Start Analysis & Elaboration**.

Натисніть кнопку ОК у вікні повідомлення, коли аналіз завершиться. Якщо були виявлені будь-які помилки, перевірте всі зв'язки схеми або повторно викличте **MegaWizard Plug-In Manager** для виправлення помилок в мегафункції. Виконайте повторну перевірку.

3. Виконайте повну компіляцію версії проекту **Lab2_SM**.

3.1. Переключіться на батьківську версію проекту **Lab2_SM**, вибравши його назву в списку, що випадає, на панелі інструментів. Якщо під час виконання Лабораторної роботи №1 перевірка проекту пройшла успішно виконайте його компіляцію, для цього виберіть команду **Start Compilation** в меню **Processing** або натисніть піктограму , розташовану на панелі інструментів. Після завершення процесу компіляції відкриється діалогове вікно, в якому натисніть кнопку **OK**.

3.2. Проаналізуйте інформацію про використані ресурси 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. 2.4), випишіть в таблицю (табл. 2.1) наступні значення: **Total logic elements, total memory bits, total embedded multiplier 9-bit elements i 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. 2.4. 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. 2.1) використані ресурси для реалізації функціональних блоків (рис. 2.5).

| 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_component1 | 1 (0) | 0 (0) |
| 4 | ladd_sub_03:auto_generated | 1 (1) | 0 (0) |

Рис. 2.5. Фрагмент розділу **Resource Utilization by Entity** звіту компілятора **Compilation Report**

- З розділу **Control Signals** звіту компілятора **Compilation Report**, випишіть основні керуючі сигнали і їх коефіцієнт розгалуження.

В даній лабораторній роботі відсутній керуючий сигнал **clock**. В звіті компілятора відображується кількість регістрів, на які надходить сигнал **clock**, коефіцієнт розгалуження, говорить про кількість задіяних архітектурних блоків: 1 блоків пам'яті, 1 блоків помножувачів, логічних комірок. Ці та інші додаткові способи аналізу проекту будуть досліджені в подальших лабораторних роботах.

- Оцініть часові затримки за результатами звіту компілятора **Timing Analyzer** із розділів **Timing Analyzer Summary** і **tpd** (рис. 2.6).

| 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. 2.6. 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 2.1.

Аналіз звіту компілятора

| Звіти компілятора | <i>Lab1_SM</i> | <i>Lab2_SM_v2</i> |
|------------------------------------|----------------|-------------------|
| Total logic elements | 2 (<1%) | |
| Total register | 0 | |
| Total memory bits | 0 | |
| Embedded multiplier 9-bit elements | 0 | |
| Total pins | 5 (3%) | |
| SM2 | 1 Logic Cells | |
| Actual Time | 9,813 ns | |
| | | |
| | | |

3.3. Виконайте аналіз логічної реалізації проекту за допомогою утиліти *RTL Viewer*

Утиліта *RTL Viewer* дозволяє представити логічну реалізацію проекту в графічному вигляді. Це інструмент для аналізу результатів синтезу *HDL* проектів. Результати представлені утилітою це результати функціонального синтезу проекту, які не є його практичною реалізацією в мікросхемі *Cyclone II*.

Викличте утиліту *RTL Viewer* із меню *Tools* із списку *Netlist Viewers*. Графічне відображення логічної реалізації проекту зображене на рис. 2.7. На схемі відображені контакти введення/виведення, блоки напівсуматорів.

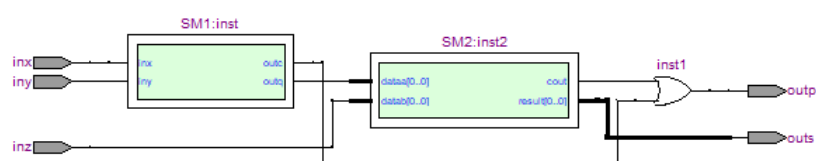


Рис. 2.7. Логічна реалізація проекту *Lab1_SM*

Виділіть блок **SM2**. Натисніть на праву кнопку миші виберіть у контекстному меню команду **Hierarchy Down** (перехід на нижчій рівень ієрархії). Відкриється внутрішня структура нижнього рівня ієрархії блоку **SM2.inst2**, з якої видно, що блок створений на базі однієї мегафункції **lpm_add_sub**, яка створена автоматично (рис. 2.8). Натисніть ще раз мишею на блоці **lpm_add_sub** (можливо доведеться опуститися на декілька рівнів) і перегляньте логічну схему блоку перетворену на примітиви **Quartus II** (рис. 2.8).

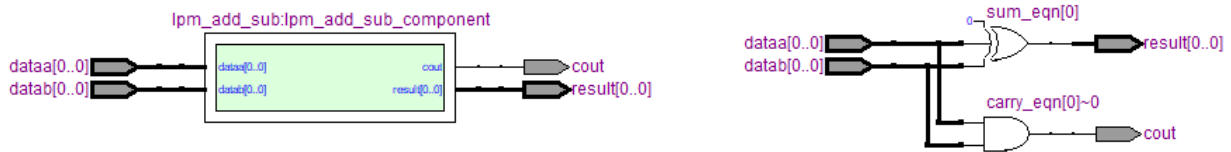


Рис. 2.8. Результати відображення логічної структури блоку **SM2** в утиліті **RTL Viewer**

Команда **Ungroup Selected Nodes** в контекстному меню корисна, якщо реалізовані багаторозрядні шини. За застосування цієї команди блоки будуть відображатись з усіма виводами. В цьому випадку видно, як поєднуються окремі сигнали. Для повернення до попереднього вигляду використайте команду **Group Related Nodes**.

Для переходу на верхні рівні ієрархії використайте команду **Hierarchy Up** в тому самому контекстному меню.

Для перегляду логічної структури напівсуматора **SM1**, використайте альтернативний спосіб – два рази натисніть на зображенні блоку лівою кнопкою миші. Відкриється нижній рівень ієрархії. Для переходу на верхній рівень, два рази натисніть мишею на вільному полі графічного вікна. Видно, що блок **SM1** реалізований користувачем на логічних елементах, зображення логічної структури блоку у примітивах **Quartus II** представлено на рис. 2.9.

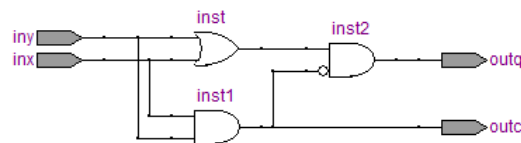


Рис. 2.9. Результати відображення логічної структури блоку **SM1** в утиліті **RTL Viewer**

Виділіть ще раз блок **SM2**. В контекстному меню оберіть команду **Display Content**.

Команда **Display Content** дозволяє відобразити нижній рівень ієрархії в межах верхнього рівня. Виділіть блок **lpm_add_sub** і в його контекстному меню оберіть команду **Display Content** – відобразиться ще один нижній рівень ієрархії. Результат використання команди **Display Content** зображено на рис. 2.10. Для закривання нижніх рівнів ієрархії використайте команду **Hide Content**.

За допомогою команди **Zoom** контекстного меню блоку проєкспериментуйте зі зміною масштабу зображення.

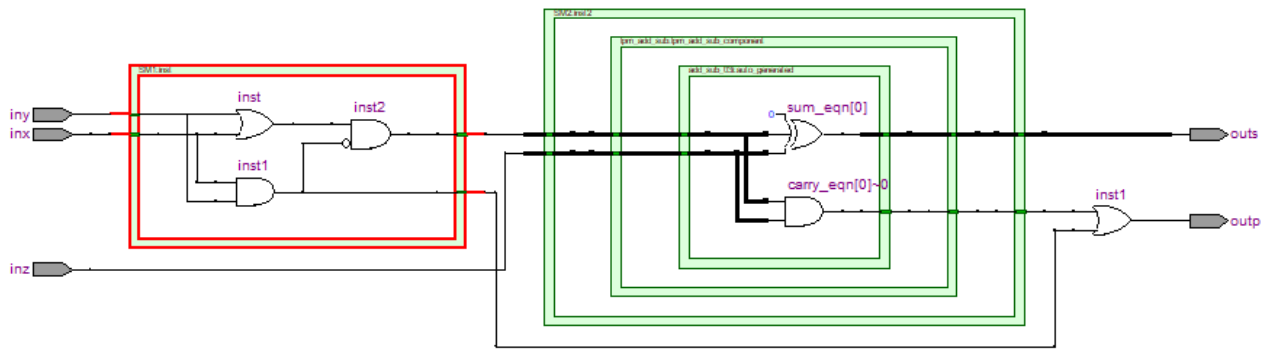


Рис. 2.10. Результат використання команди *Display Content* утиліти *RTL Viewer*

Зверніть увагу, що вікно ієрархії проекту *Hierarchy List* утиліти *RTL Viewer* також відображує ієрархічну структуру кожного блоку. Для кожного рівня ієрархії додатково відображаються імена контактів і внутрішніх мереж (рис. 2.11).

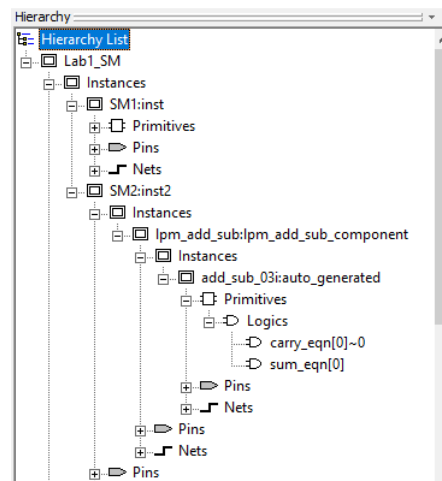
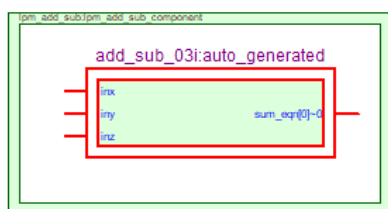


Рис. 2.11. Відображення інформації у вікні *Hierarchy List* утиліти *RTL Viewer*

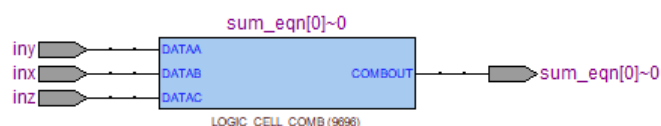
3.4. Виконайте аналіз фізичної реалізації проекту за допомогою утиліти *Technology Map Viewer*

Утиліта *Technology Map Viewer* дозволяє переглядати фактичну реалізацію проекту і використовувані ресурси FPGA/CPLD. Утиліта *Technology Map Viewer* використовується в якості допоміжного інструменту в процесі налагодження проекту для аналізу змін використаних ресурсів або змін налаштувань проекту.

Перегляньте фізичну реалізацію блоку *SM2* за допомогою утиліти *Technology Map Viewer*, для цього (І спосіб) у вікні утиліти *RTL Viewer* перейдіть на ієрархічний рівень, що відображує модуль мегафункції *lpm_add_sub*, виділіть його і оберіть у контекстному меню команду **Locate**, а далі команду **Locate in Technology Map Viewer**. В результаті відкриється вікно утиліти *Technology Map Viewer* із зображенням виділеного блоку (рис. 2.12, а). Перейдіть на нижній рівень ієрархії будь яким способом, що описаний вище (рис.2.12, б).



а



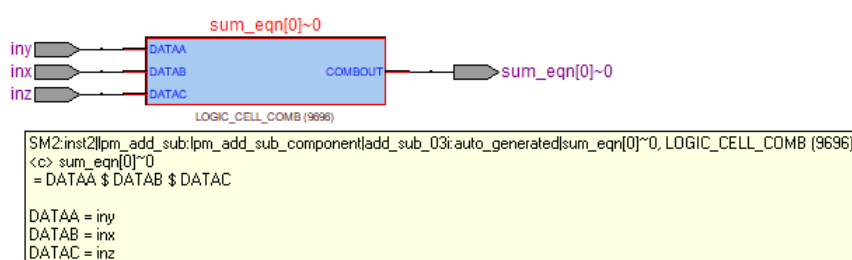
б

Рис. 2.12. Фізична реалізацію блоку **SM2** в технологічному редакторі **Technology Map Viewer**

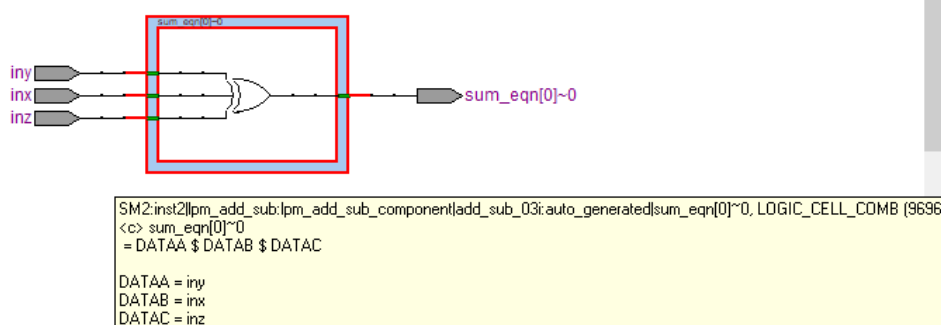
На відміну від примітивів, що зображуються на нижньому рівні утилітою **RTL Viewer**, **Technology Map Viewer** відображає фактично використаний ресурс мікросхеми – номер логічної комірки, яка використана на мікросхемі для реалізації логічного елемента, логічну функцію, яка реалізована даним логічним елементом (рис. 2.13, а).

В інших структурах утиліта **Technology Map Viewer** надає можливість переглянути будь які фактично використані ресурси, наприклад, регістри із складу логічної комірки, блоки убудованої пам'яті, помножувачі інші спеціалізовані убудовані блоки мікросхем обраних сімейств.

Двічі натисніть ліву кнопку миші на блоці **sum** для докладного аналізу реалізації модуля **sum** на базі мегафункції **lpm_add_sub** (рис. 2.13, б).



а



б

Рис. 2.13.

Результати аналізу функціонального і фізичного синтезу показують, що трасувальник виконав певну оптимізацію, що називається упаковуванням. Трасувальник перемістив логічний елемент АБО в блок напівсуматора, що реалізований за допомогою мегафункції **lpm_add_sub** для покращення

продуктивності і зменшення кількості використаних логічних ресурсів мікросхеми (в утиліті видно, що логічний елемент АБО розташовувався за межами блоку SM2). Оптимізований модуль реалізований за допомогою логічного елементу ВИКЛЮЧНЕ АБО з трьома входами. Це однорівнева комбінаційна схема, яка забезпечує збільшення швидкодії у порівнянні з дворівневою реалізацією.

В даному випадку трасувальник видає повідомлення "**Extra Info**" (додаткова інформація) у вікні повідомлень **Message** (або в таблиці **Suppressed**). Це вказує на те, що відбулася оптимізація. Це можна перевірити також за допомогою редактора ресурсів **Resource Property Editor** в команді **Locate** контекстного меню певного блоку.

Перегляньте фізичну реалізацію проекту **Lab1_SM** цілком **SM2** за допомогою утиліти **Technology Map Viewer** у інший спосіб (II спосіб), для цього викличте утиліту **Technology Map Viewer** із меню **Tools** із списку **Netlist Viewers**. Графічне відображення фізичної реалізації проекту зображене на рис. 2.14, з якого видно, що трасувальник дійсно застосував певну оптимізацію і схема відрізняється від логічної структури проекту, яка зображена на рис. 2.7.

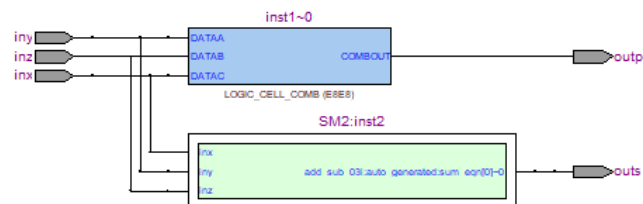


Рис. 2.7. Фізична реалізація проекту **Lab1_SM**

Дослідить фізичну структуру нижніх рівнів ієрархії блоку **SM2** у технологічному редакторі **Technology Map Viewer**.

Перегляньте фізичну реалізацію блоку **SM1** за допомогою утиліти **Technology Map Viewer**. Аналогічно до описаної вище послідовності кроків перегляньте спосіб фізичної реалізації блоку **inst1**, який є результатом оптимізації напівсуматора **SM1** трасувальником. Результат аналізу представлений на рис. 2.14.

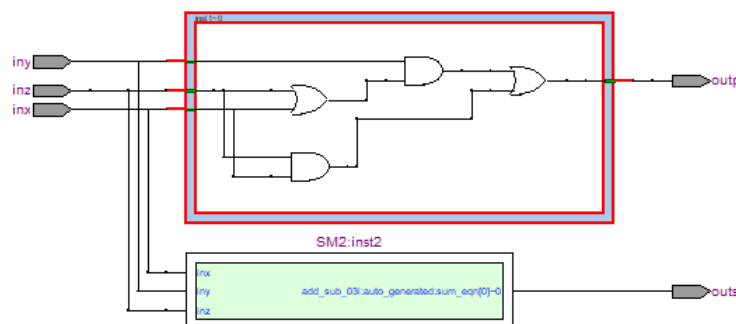



Рис. 2.8. Фізична структура оптимізованого блоку напівсуматора в технологічному редакторі **Technology Map Viewer**.

4. Перевірте зв'язки в проекті за допомогою редактора топології кристала **Chip Planner**.

Редактор **Chip Planner** надає можливість перегляду фізичного розміщення логічних

комірок вашого проекту на кристалі. Редактор *Chip Planner* зображує особливості реалізації проекту, а також архітектурні особливості FPGA/CPLD. Редактор *Chip Planner* призначений для ручного призначення логічних ресурсів, але може бути використаний й для оцінки отриманих результатів компіляції.

4.1. У вікні технологічного редактора *Technology Map Viewer*, виділіть блок *SM2:inst2*. Натисніть праву кнопку миші і виберіть команду **Locate** \Rightarrow **Locate in Chip Planner (Floorplan & Chip Editor)** в контекстному меню. У вікні редактора топології кристалу *Chip Planner*, що відкрилося, оберіть піктограму  на панелі інструментів, і за допомогою правої кнопки миші зменшить зображення в декілька разів. На зображенні, що відкрилося визначений блок виділений синім кольором.

Всі інші задіяні ресурси (або, якщо зняти виділення) будуть виділені коричневим кольором. В середньому полі знаходяться логічні комірки, що угрупованні в логічні блоки по 2x8 логічних комірок. Вільні комірки зображені білим кольором. У вертикальних рядах розміщені убудовані блоки *RAM* і помножувачів. По периметру – контакти (піни) мікросхеми.

Огляньте зображення кристалу у редакторі. Видно, що зайнято дві логічні комірки і задіяно п'ять пінів, що відповідає звіту компілятора. Позначення задіяних пінів у вікні редактора відповідає позначенням, що записано в звіті компілятора (*Compilation Report* > *Resource Section* > *Input Pins (Output Pins)*) (рис. 2.11).

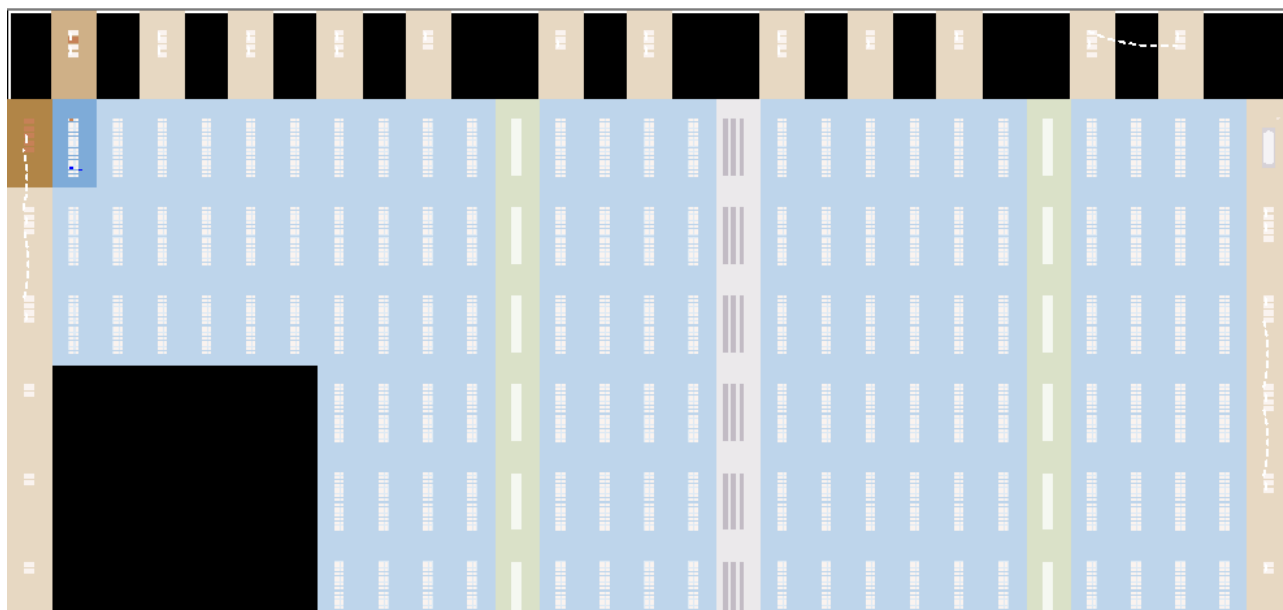


Рис. 2.9. Розміщення логічних ресурсів на мікросхемі *Cyclon II*.

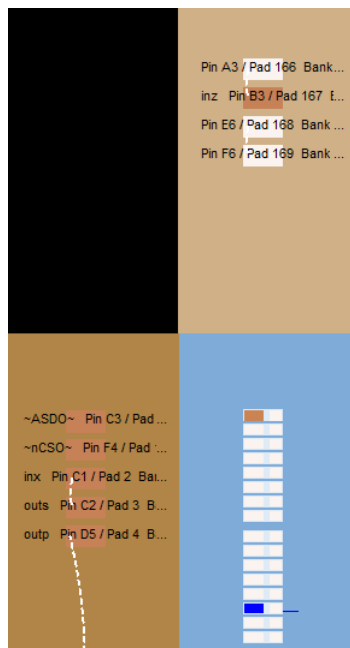


Рис. 2.10. Збільшений фрагмент кристалу у редакторі *Chip Planner*

| Input Pins | | | | | | | |
|------------|------|-------|----------|--------------|--------------|-------------|-----------------------|
| | Name | Pin # | I/O Bank | X coordinate | Y coordinate | Cell number | Combinational Fan-Out |
| 1 | inx | C1 | 1 | 0 | 13 | 2 | 2 |
| 2 | iny | N8 | 4 | 7 | 0 | 2 | 2 |
| 3 | inz | B3 | 2 | 1 | 14 | 1 | 2 |

| Output Pins | | | | | | | |
|-------------|------|-------|----------|--------------|--------------|-------------|---|
| | Name | Pin # | I/O Bank | X coordinate | Y coordinate | Cell number | |
| 1 | outp | D5 | 1 | 0 | 13 | 4 | t |
| 2 | outs | C2 | 1 | 0 | 13 | 3 | t |


Рис. 2.11. Звіт компілятора щодо задіяних контактів мікросхеми

У вікні редактора *Chip Planner* виділені області – це ресурси, що використовуються для розміщення логічних блоків проекту на мікросхемі *Cyclone II*. У випадку, якщо користувачем не зроблено ніяких призначень щодо розміщення логічних блоків, трасувальник самостійно призначає необхідні ресурси.

4.2. Якщо блок пам'яті виділено у вікні *Chip Planner*, натисніть кнопку



(**Generate Fan-In Connections**) на панелі інструментів (відображення вхідних зв'язків для виділеного блоку). У вікні *Chip Planner* відобразяться ті логічні ресурси мікросхеми, від яких надходять вхідні сигнали до логічного блоку (блоки введення/вивення) і затримки поширення сигналів по виділеним зв'язкам.

Виділіть знову блок логічний блок *SM2* в *Chip Planner* (за необхідності зменшить масштаб зображення) і натисніть кнопку  (**Generate Fan-Out Connections**) на панелі інструментів. У вікні *Chip Planner* одночасно відобразяться вхідні і вихідні зв'язки виділеного блоку (рис. 2.12). Вихідні зв'язку показані синім кольором.

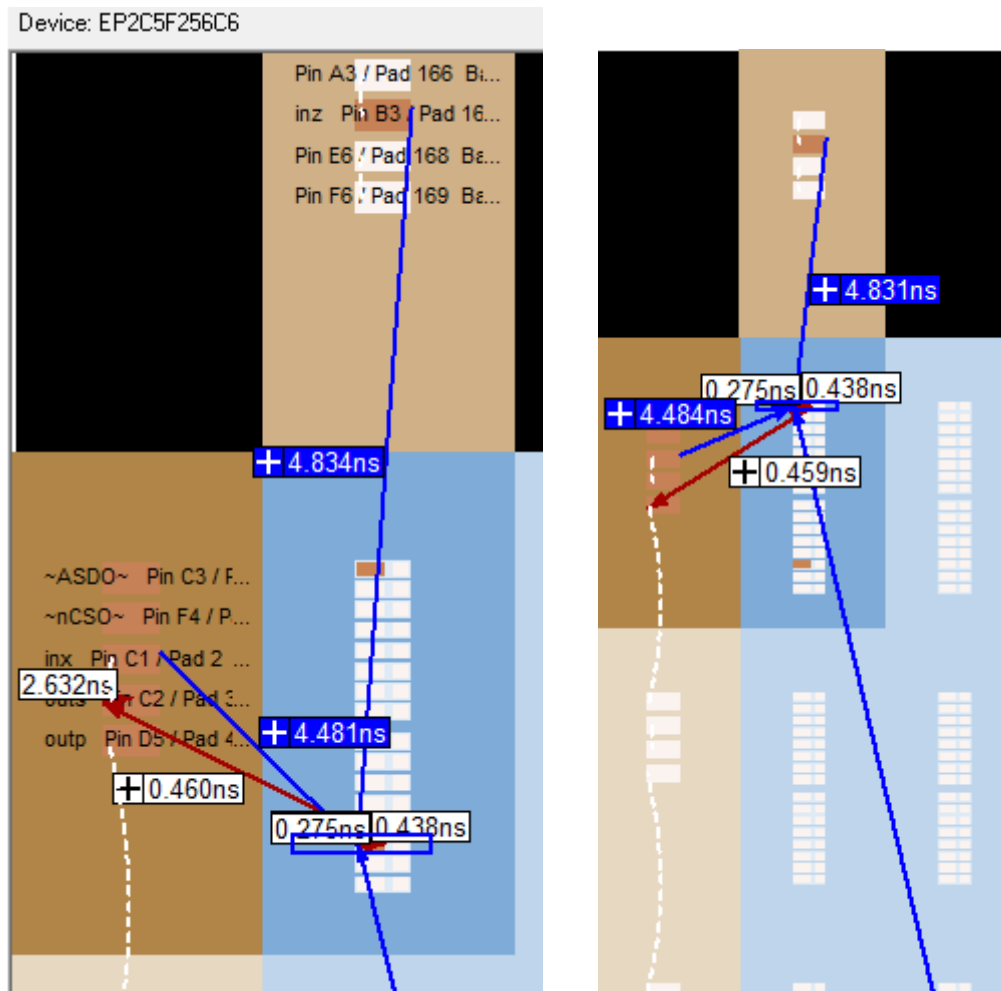

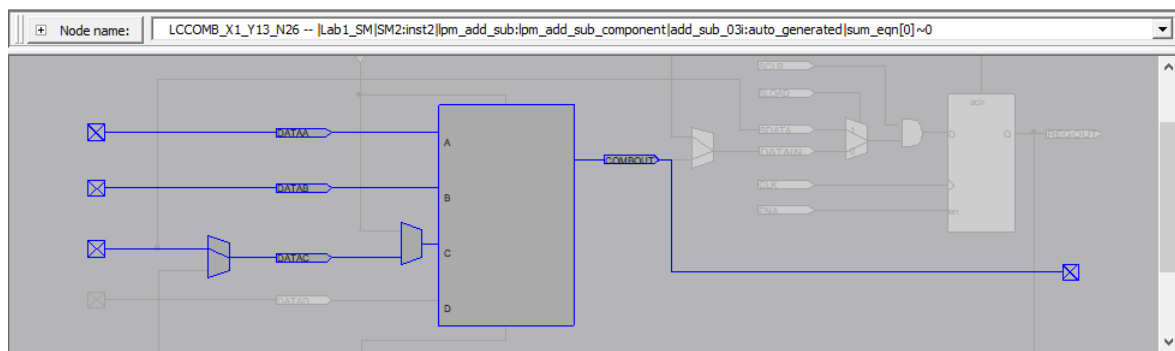


Рис. 2.12. Вхідні і вихідні зв'язки в редакторі *Chip Planner*

Скасуйте команди **fan-in/fan-out**, натиснувши лівою кнопкою миші на будь-якому невиділеному блоці, а потім натисніть кнопку  (**Clear Unselected Connections / Paths**) на панелі інструментів.

4.3. Перегляньте використані ресурси логічних комірок, які задіяні для реалізації комбінаційних схем. Для цього натисніть два рази лівою кнопкою миші на виділеному блоці. При цьому відкриється вікно редактора ресурсів *Resource Property Editor* (рис. 2.13).



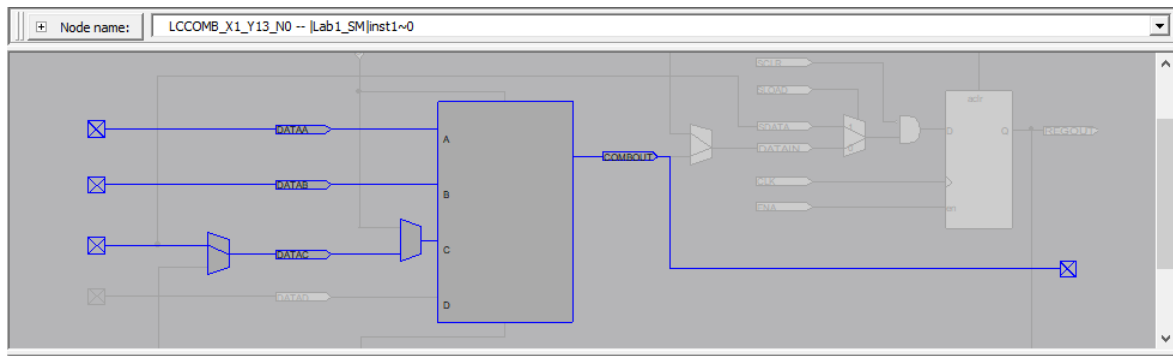


Рис. 4.3. Аналіз використаних ресурсів логічних комірок у редакторі ресурсів *Resource Property Editor*

4.4. Зверніть увагу, що редактор топології кристалу *Chip Planner* можна відкрити із меню *Tools* із списку *Netlist Viewers*.

5. Закрийте вікна *RTL Viewer*, *Technology Map Viewer* та *Chip Planner*.

Індивідуальне завдання

1. Намалюйте в протоколі вихідну схему проекту *Lab1_SM*, яка була введена в САПР, логічну схему в примітивах *Quartus II* і схему фізичної реалізації проекту на визначеному сімействі мікросхем, виконану САПР *Quartus II*, на базі отриманих результатів компіляції проекту і перегляду результатів в утилітах *RTL Viewer* і *Technology Map Viewer*.
2. Випишіть із звіту компілятора *Compilation Report* час розповсюдження сигналів за всіма ланцюгами схем (рис. 2.6). Порівняйте схеми і зробіть висновки, щодо оптимізації, виконаної трасувальником.
3. Зробіть висновки, щодо способу реалізації схеми напівсуматора – користувацька схема, або мегафункція розроблена компанією Altera. Оцініть швидкодію роботи блоків і використані ресурси.
4. Виконайте повну компіляцію другої версії проекту *Lab2_SM_v2*.
5. Виконайте аналіз результатів компіляції за всіма кроками, що описані вище.
6. Доповніть табл. 2.1 отриманими параметрами.
7. Додайте до схем, намальованих в протоколі (згідно пункту 1 індивідуального завдання) схеми реалізації повного суматора за допомогою мегафункції Altera.
8. Випишіть із звіту компілятора *Compilation Report* час розповсюдження сигналів за всіма ланцюгами схем (рис. 2.6). Порівняйте схеми і зробіть висновки, щодо доцільності оптимізації, виконаної трасувальником.
9. Зробіть висновки, щодо способу реалізації схеми напівсуматора – користувацька схема, або мегафункція розроблена компанією Altera. Оцініть швидкодію роботи блоків і використані ресурси.
10. Зробіть загальні висновки