

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Запорізький національний технічний університет

МЕТОДИЧНІ ВКАЗІВКИ
до практичних занять з дисципліни
**”Комп'ютерне моделювання та проектування
пристроїв цифрової електроніки”**
для студентів спеціальності 6.050801
„Мікро- та наноелектроніка”
денної і заочної форм навчання

Методичні вказівки до практичних занять з дисципліни ”Комп'ютерне моделювання та проектування пристроїв цифрової електроніки” для студентів спеціальності 6.050801 “Мікро- та наноелектроніка” денної і заочної форм навчання / Укл.: Н.М. Нагорна. –Запоріжжя: ЗНТУ, 2011.- 82 с.

Укладач: Н.М. Нагорна, ст.викладач

Рецензент: О.В.Томашевський, доц., канд.техн. наук

Відповідальний за випуск: Г.В. Сніжной, доц., канд.фіз.-матем. наук

Затверджено
на засіданні кафедри
“Мікро- та наноелектроніка”

Протокол № 4
від 1 лютого 2011 р.

1. ЗМІСТ

1 Практичне заняття №1 "Моделі послідовнісних схем".....	4
2 Практичне заняття №2 "Проектування синхронних схем на основі моделей Мура і Мілі".....	8
3 Практичне заняття №3 "Описи об'єктів на мові VHDL".....	15
4 Практичне заняття №4 "Реалізація автоматних VHDL-моделей".....	28
5 Практичне заняття №5 "Проектування електронних схем на ПЛІС".....	36
6 Практичне заняття №6 "Розрахунок параметрів і синтез топології планарного дрейфового n - p - n транзистора".....	42
7 Практичне заняття №7 "Розрахунок параметрів і синтез топології ЛЕ АБО-НЕ на МОН-транзисторах".....	57
8 Перелік рекомендованої літератури.....	73
Додаток А VHDL–код, що реалізує алгоритм роботи системи S	74
Додаток Б Автомат Мілі з п'ятьма станами.....	76
Додаток В Варіанти завдань до практичного заняття № 6.....	77
Додаток Г Варіанти завдань до практичного заняття № 7.....	80

1 ПРАКТИЧНЕ ЗАНЯТТЯ №1

"МОДЕЛІ ПОСЛІДОВНІСНИХ СХЕМ"

Метою роботи є: ознайомлення з формальним підходом до опису і проектування послідовнісних схем довільного типу, розгляд прикладів проектування схем з використанням моделі Мура.

1.1 Теоретичні відомості

1.1.1 Моделі синхронної послідовнісної схеми

У послідовнісних схемах існує кінцева кількість логічних станів, тому вони називаються кінцевими автоматами. Стан послідовнісних схем представляється двійковими сигналами, які називаються змінними станів.

Розглянемо модель Мілі (Mealy model), яка є моделлю послідовнісної логічної схеми на D -тригерах (рис. 1.1). У моделі можуть бути використані також інші типи тригерів.

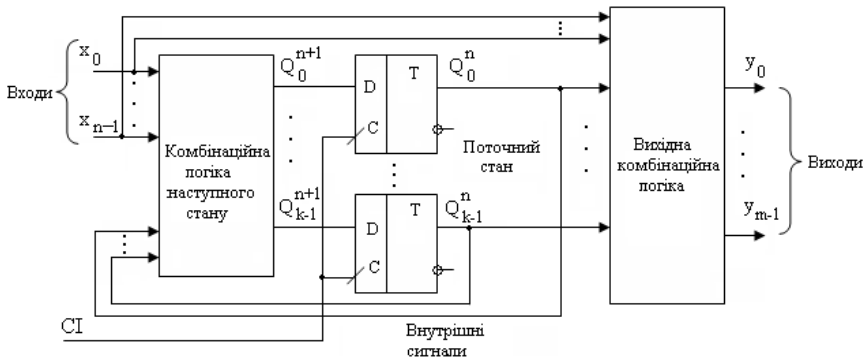


Рисунок 1.1 – Модель Мілі

У моделі Мілі сигнали Q^n визначають поточний стан схеми. Це змінні поточних станів. Сигнали Q^{n+1} – змінні наступних станів.

Сигнали Q^{n+1} – це комбінаційні функції вхідних сигналів X і поточних станів Q^n .

Особливості моделі: поточні стани Q^n змінюються у момент надходження активуючого переходу тактового імпульсу C . Тому зміна вхідного сигналу X не впливатиме на Q^n при відсутності імпульсу C .

Вихідні сигнали Y описуються комбінаційними функціями вхідних сигналів і поточних станів. Тому зміни Y можуть відбуватися при зміні X незалежно від імпульсу C .

Цієї проблеми легко уникнути, якщо сигнали Y залежатимуть тільки від поточних станів Q^n .

Подібний підхід реалізується в моделі Мура (Moore model), наведеної на рис. 1.2.

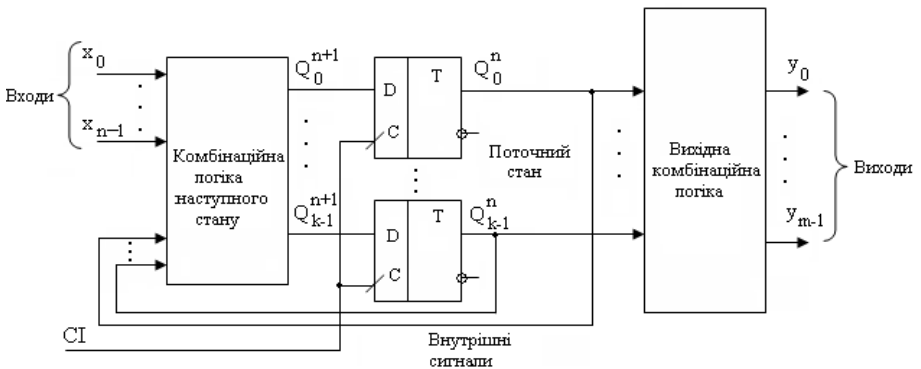


Рисунок 1.2 - Модель Мура

При проектуванні послідовних схем передусім необхідно визначити кількість тригерів у схемі. У загальному випадку, при кількості станів від $2^{k-1}+1$ до 2^k необхідно мати k змінних стану і, отже, k тригерів.

Якщо використовуються тригери з перемиканням по позитивному фронту, то всі зміни станів відбуватимуться під час переходів рівня тактового сигналу з 0 в 1. Якщо ж застосовуються тригери з перемиканням по негативному фронту, то зміни відбуватимуться по задньому фронту тактового сигналу, тобто при переході з 1 в 0.

У будь-якому випадку, після приходу активуючого переходу тактового імпульсу необхідно забезпечити достатню затримку до моменту надходження переходу наступного тактового імпульсу, щоб сиг-

нали встигли пройти схеми комбінаційної логіки і встановити значення змінних наступного стану.

Проектування послідовної синхронної схеми можна виконувати як з використанням моделі Мілі, так і з використанням моделі Мура, але в моделі Мілі зміни входніх сигналів можуть впливати на стан виходів безпосередньо у момент зміни, а не синхронно з приходом тактового імпульсу. Тому в цьому аспекті поведінка моделей може відрізнятися.

Лічильники – це схеми, поведінка яких описується моделлю Мура, оскільки виходи схеми – це безпосередньо виходи тригерів.

При проектуванні послідовних схем необхідно визначити функції змінних наступних станів, які визначаються вектором Q^{n+1} , і функції вихідних станів, які визначаються вектором Y .

1.1.2 Проектування схеми лічильника на базі моделі Мура

На прикладі задачі 1 розглянемо проектування схеми з використанням моделі Мура.

Задача 1. Розробити лічильник на базі D -тригерів із двома довільними рахунковими послідовностями.

Нехай A – вхід керування. При $A = 0$ формується послідовність 00, 01, 11. При $A = 1$ – послідовність 00, 11, 01.

Розв'язок. На основі умови задачі будується граф станів моделі Мура для лічильника. Кожна із заданих послідовностей має три стани, але при вмиканні схеми може виникнути і четвертий стан.

На графі станів лічильника (рис. 1.3) у вузлах вказані номери станів і значення сигналів на виходах схеми; над дугами проставлені значення сигналів керування A , необхідні для переходу із одного стану в інший. У даному випадку номери станів вибрані довільно.

На основі графа станів будується таблиця станів (табл. 1.1).

Далі станам призначаються значення змінних, тобто для кожного стану

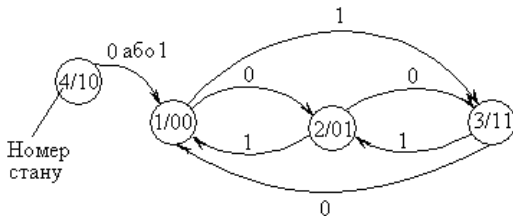


Рисунок 1.3 - Граф станів лічильника

вибраються значення змінних поточних станів. Для зручності стану 1 привласнюється комбінація значень змінних $Q_1^n Q_0^n = 00$; стану 2 - $Q_1^n Q_0^n = 01$; стану 3 - $Q_1^n Q_0^n = 11$; стану 4 - $Q_1^n Q_0^n = 10$. Вказане привласнення відповідає вихідним станам, які вимагаються в умові задачі. Результати призначень зведені в табл. 1.2.

Таблиця 1.1 - Таблиця станів лічильника

Поточний стан Q^n	Наступний стан Q^{n+1}	
	$A = 0$	$A = 1$
1	2	3
2	3	1
3	1	2
4	1	1

Таблиця 1.2 - Таблиця призначених станів

Поточний стан $Q_1^n Q_0^n$	Наступний стан $Q_1^{n+1} Q_0^{n+1}$	
	$A = 0$	$A = 1$
00	01	11
01	11	00
11	00	01
10	00	00

Далі слід визначити вхідні функції тригерів з використанням табл. 1.2. Карти Карно для обох вхідних функцій та отримані на їх основі вказані функції наведені на рис. 1.4.

$Q_1^n Q_0^n \backslash A$	00	01	11	10
0	0	1	0	0
1	1	0	0	0

$$D_1 = Q_1^{n+1} = \overline{Q_1^n} Q_0^n \overline{A} + \overline{Q_1^n} Q_0^n A$$

$Q_1^n Q_0^n \backslash A$	00	01	11	10
0	1	1	0	0
1	1	0	1	0

$$D_0 = Q_0^{n+1} = \overline{Q_1^n} \overline{A} + \overline{Q_1^n} Q_0^n A + Q_1^n Q_0^n A$$

Рисунок 1.4 – Карти Карно і мінімізовані вхідні функції тригерів

Мінімізація функції D_0 неповна. Член $\overline{Q_1^n} \overline{Q_0^n} A$ вибраний тому, що він зустрічається в D_1 .

1.2 Завдання

1.2.1 На основі одержаної в задачі 1 з пункту 1.1.2 моделі лічильника синтезуйте його логічну структуру.

1.2.2 Розробіть логічну структуру лічильника із рахунковою послідовністю 11, 01, 10.

1.3 Зміст звіту

Звіт повинен містити: мету роботи; моделі Мілі та Мура; схему лічильника із задачі 1; граф станів лічильника із рахунковою послідовністю 11, 01, 10; таблиці станів і призначених станів цього лічильника; вхідні функції тригерів; схему лічильника.

1.4 Контрольні запитання

- 1 Чи можна в моделях Мілі та Мура використовувати *JK*-тригери?
- 2 Скільки тригерів буде в схемі лічильника із 48 станами?
- 3 Чим відрізняються моделі Мілі та Мура?
- 4 Чи може бути призначення станам значень змінних довільним?
- 5 Коли доцільна неповна мінімізація вхідних функцій тригерів?
- 6 Чим відрізняється комбінаційна логіка від послідовної?

2 ПРАКТИЧНЕ ЗАНЯТТЯ №2

"ПРОЕКТУВАННЯ СИНХРОННИХ СХЕМ НА ОСНОВІ МОДЕЛЕЙ МУРА І МІЛІ"

Метою роботи є: ознайомлення із алгоритмом синтезу послідовних схем та з підходами до кодування станів; розгляд конкретних реалізацій алгоритму з використанням моделей Мура та Мілі.

2.1 Теоретичні відомості

2.1.1 Алгоритм проектування електронної схеми

Алгоритм синтезу послідовної схеми містить наступні кроки:

- побудова діаграми станів;
- побудова таблиці станів;
- присвоєння станам схеми комбінацій значень змінних стану;

- побудова таблиці призначених станів;
- отримання входніх функцій тригерів (за допомогою карт Карно);
- отримання за картами Карно вихідних функцій;
- побудова логічної схеми.

Реалізацію наведеного алгоритму розглянемо на прикладі задачі.

Задача. Синтезувати схему, яка детектує певну кодову послідовність, що надходить на вхід даних. Логічний стан цього входу може змінюватися після кожного тактового імпульсу. Схема має один вихід, стан якого буде рівним 1 тільки тоді, коли на вході з'являється задана послідовність (рис. 2.1).

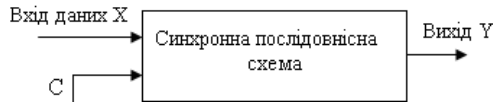


Рисунок 2.1 – Узагальнений вид детектора двійкової послідовності

Нехай, потрібно визначити наявність на вході кодової послідовності 0110011. Причому стан на виході Y має встановлюватися в 1 при появі на вході останнього біта послідовності.

Розв'язок. Послідовність можна реалізувати за 8 кроків, що передбачає наявність восьми станів. При побудові графа станів (рис. 2.2) використана модель Мура, в якій кожному стану відповідає конкретне значення вихідного сигналу Y .

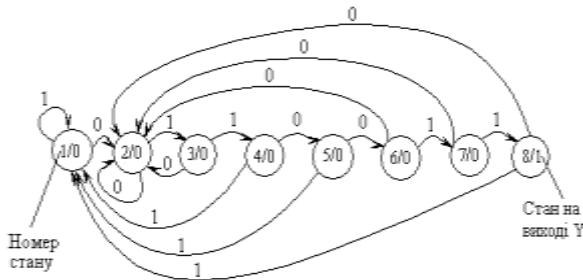


Рисунок 2.2 – Граф станів детектора двійкової послідовності

Початковим станом є стан 1, в якому $Y = 0$. Схема змінює свій стан, якщо на вході з'являється 0 (точніше, якщо сигнал 0 на вході існує під час приходу тактового імпульсу).

Це перший 0 послідовності, який потрібно знайти. Після його появи схема переходить у стан 2. При цьому на виході зберігається значення 0. Поява наступних станів вхідної послідовності приводить до переходу схеми в стани 3, 4, 5, 6, 7 і 8. У останньому восьмому стані на виході схеми встановлюється 1.

Після цього наступна 1 повертає схему в початковий стан із номером 1. Якщо ж приймається 0, то він може бути першим нулем нової послідовності, тому виконується перехід до стану 2.

Вся решта переходів в графі станів, що мають місце до прийому останнього біта послідовності, є переходами у стан 1, якщо приймається 1, і у стан 2, якщо приймається 0 (який розглядається як перший нуль послідовності).

Таблицю станів (переходів) детектора наведено в табл. 2.1.

Таблиця 2.1 - Таблиця станів детектора послідовності

Поточний стан	Наступний стан		Вихідний стан Y
	X=0	X=1	
1	2	1	0
2	2	3	0
3	2	4	0
4	5	1	0
5	6	1	0
6	2	7	0
7	2	8	0
8	2	1	1

Призначення станам значень змінних. Для опису восьми різних станів потрібні 3 змінні стану, нехай Q_3, Q_2, Q_1 . Необхідно кожному із станів призначити певні значення змінних. Призначення може бути довільним.

Існує безліч варіантів таких призначень. Оптимального способу вибору варіанту не існує. Простішу функцію можна знайти, якщо на карті Карно є клітини із суміжними 1.

При призначенні можна використовувати нижченаведені правила.

Правило №1. При переході в наступний однаковий стан призначаються коди, відмінні значеннями тільки однієї змінної.

Правило №2. Наступним станам, в які може переходити поточний стан, слід призначати коди із однією змінною, що розрізняється.

Правило №3. Станам із однаковим значенням на виході слід призначати коди, відмінні значеннями тільки однієї змінної.

Правило №3 застосовують тоді, коли не можна використовувати правила №1 і №2.

Ілюстрації до наведених правил показані на рис. 2.3.



Рисунок 2.3 - Ілюстрації до правил №1, №2, №3

Указані правила є рекомендаціями, і не можуть гарантувати отримання схеми із мінімальною кількістю логічних елементів.

Для даної задачі, застосувавши правило №1 для станів 1, 4, 5 і 8 (всі вони переходять у стан 1) і станів 3, 6, 7 (вони переходять в один і той же стан 2), можна отримати наступні призначення:

стан 1	000;	стан 2	111;
стан 4	001;	стан 3	101;
стан 5	010;	стан 6	110;
стан 8	100;	стан 7	011.

Присвоєння стану 1 коду 000 полегшує ініціалізацію тригерів у цьому стані (за допомогою входів скиду).

У табл. 2.2 наведено коди призначених станів.

Таблиця 2.2 - Таблиця призначених станів детектора

Поточний стан $Q_3^n Q_2^n Q_1^n$	Наступний стан $Q_3^{n+1} Q_2^{n+1} Q_1^{n+1}$		Значення на виході Y
	$X = 0$	$X = 1$	
000	111	000	0
111	111	101	0
101	111	001	0
001	010	000	0
010	110	000	0
110	111	011	0
011	111	100	0
100	111	000	1

Вид вихідної функції детектора нескладно визначити із останнього рядка табл. 2.2 за кодом поточного стану:

$$Y = Q_3^n \overline{Q_2^n} \overline{Q_1^n}.$$

Функції збудження тригерів визначають за допомогою карт Карно.

$Q_2^n Q_1^n \backslash X Q_3^n$	00	01	11	10
00	1	0	1	1
01	1	1	1	1
11	0	0	1	0
10	0	0	1	0

$$D_3 = Q_3^{n+1} = \overline{X} \overline{Q_1^n} + \overline{X} Q_3^n + Q_2^n Q_1^n$$

$Q_2^n Q_1^n \backslash X Q_3^n$	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	0	0	0	1
10	0	0	0	0

$$D_2 = Q_2^{n+1} = \overline{X} + Q_3^n \overline{Q_2^n} \overline{Q_1^n}$$

$Q_2^n Q_1^n \backslash X Q_3^n$	00	01	11	10
00	1	0	1	0
01	1	1	1	1
11	0	1	1	1
10	0	0	0	0

$$D_1 = Q_1^{n+1} = Q_3^n Q_1^n + Q_3^n Q_2^n + \overline{X} \overline{Q_2^n} \overline{Q_1^n} + \overline{X} Q_2^n Q_1^n$$

Користуючись отриманими вище функціями, можна зобразити схему детектора послідовності.

2.1.2 Проектування схем на основі моделі Мілі

Модель Мілі відрізняється від моделі Мура тим, що кожному стану не обов'язково відповідає один набір вихідних значень, тому вихідний сигнал залежить як від поточного стану, так і від сигналів на входах, що перемикають схему в даний стан.

На графі станів Мілі над дугами, спрямованими до стану, вказуються як вихідні, так і вхідні сигнали, які призводять до переходу в даний стан.

Часто для вирішення однієї і тієї ж задачі граф станів моделі Мілі має менше станів, ніж аналогічний граф моделі Мура, тому змінних стану і тригерів менше. При цьому вихідні функції можуть мати складнішу форму, тому що в них використовуються як змінні стану, так і вхідні змінні.

Суттєво те, що у випадку зміни вхідних сигналів в проміжках між надходженням тактових імпульсів можуть змінюватися значення виходів, що може впливати на схемні рішення системи в цілому.

Нижче приводиться алгоритм рішення задачі пункту 2.1.1 (синтез детектора двійкової послідовності 0110011) з використанням моделі Мілі. Граф станів моделі Мілі показаний на рис. 2.4.

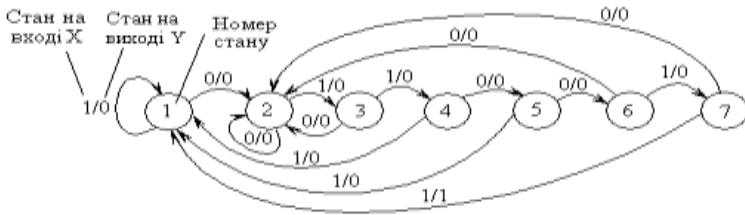


Рисунок 2.4 - Граф станів моделі Мілі для детектора двійкової послідовності

Таблиці станів моделі Мілі і призначених станів наведено в табл. 2.3 і табл. 2.4.

У табл. 2.4 стан 100 не використовується, він ніколи не з'явиться, якщо при ввімкненні всі тригери обнулити спільним сигналом скиду. У табл. 2.4 використано ті ж призначення, що і в моделі Мура.

Таблиця 2.3 - Таблиця станів детектора

Поточний стан	Наступний стан		Наступний вихідний стан Y	
	X = 0	X = 1	X = 0	X = 1
1	2	1	0	0
2	2	3	0	0
3	2	4	0	0
4	5	1	0	0
5	6	1	0	0
6	2	7	0	0
7	2	1	0	1

Таблиця 2.4 - Таблиця призначених станів детектора

Поточний стан $Q_3^n Q_2^n Q_1^n$	Наступний стан $Q_3^{n+1} Q_2^{n+1} Q_1^{n+1}$		Наступний вихідний стан Y	
	$X = 0$	$X = 1$	$X = 0$	$X = 1$
000	111	000	0	0
111	111	101	0	0
101	111	001	0	0
001	010	000	0	0
010	110	000	0	0
110	111	011	0	0
011	111	000	0	1

Отримана на основі табл. 2.4 вихідна функція має вигляд:

$$Y = X \overline{Q_3^n} Q_2^n Q_1^n.$$

Функції збудження тригерів визначають за допомогою карт Карно.

$Q_2^n Q_1^n$ XQ_3^n	$Q_2^n Q_1^n$				$Q_2^n Q_1^n$ XQ_3^n	$Q_2^n Q_1^n$				$Q_2^n Q_1^n$ XQ_3^n	$Q_2^n Q_1^n$			
	00	01	11	10		00	01	11	10		00	01	11	10
00	1	0	1	0	00	1	1	1	1	00	1	0	1	1
01	*	1	1	1	01	*	1	1	1	01	*	1	1	1
11	*	1	1	1	11	*	0	0	1	11	*	0	1	0
10	0	0	0	0	10	0	0	0	0	10	0	0	0	0

$$D_1 = Q_1^{n+1} = Q_3^n + \overline{X} \overline{Q_2^n} \overline{Q_1^n} + \overline{X} Q_2^n Q_1^n \quad D_2 = Q_2^{n+1} = \overline{X} + Q_3^n \overline{Q_1^n} \quad D_3 = Q_3^{n+1} = \overline{X} \overline{Q_1^n} + \overline{X} Q_3^n + Q_3^n Q_2^n Q_1^n + \overline{X} Q_2^n$$

2.2 Завдання

2.2.1 З використанням розробленої у задачі пункту 2.1.1 на основі автомата Мура моделі детектора двійкової послідовності синтезуйте його логічну структуру.

2.2.2 З використанням розробленої у пункті 2.1.2 на основі автомата Мілі моделі детектора двійкової послідовності синтезуйте його логічну структуру.

2.2.3 На основі автомата Мілі реалізуйте модель детектора двійкової послідовності 10110 і синтезуйте його логічну структуру.

2.3 Зміст звіту

Звіт повинен містити: мету роботи; алгоритм синтезу послідовної схеми; логічну структуру детектора двійкової послідовності 0110011 на основі автомата Мура (модель синтезована у задачі пункту 2.1.1); логічну структуру детектора двійкової послідовності 0110011 на основі автомата Мілі (модель синтезована у пункті 2.1.2); граф станів детектора двійкової послідовності 10110; таблиці станів і призначених станів детектора; вхідні функції тригерів; схему детектора.

2.4 Контрольні запитання

- 1 Задані чотири стани: 1, 2, 3, 4, для яких відомо, що стан 1 переходить у стан 2, стан 2 переходить у стан 3, стан 3 переходить у стан 4, стан 4 переходить у стан 1. Першому стану привласнюється код 00. Призначте значення змінних стану двома різними способами.
- 2 У чому різниця між графами станів моделі Мура і моделі Мілі?

3 ПРАКТИЧНЕ ЗАНЯТТЯ №3 "ОПИСИ ОБ'ЄКТІВ НА МОВІ VHDL"

Метою роботи є: ознайомлення з видами опису систем на структурному і поведінковому рівнях; ознайомлення з представленням опису системи у вигляді дерев ієрархії різних типів; розгляд двох типів описів об'єкта мовою VHDL: опис об'єкта "у цілому" (*entity*) і опис архітектури об'єкта (*architecture*); ознайомлення з типами сигналів.

3.1 Види опису цифрової системи

VHDL - це могутня мова, яка дозволяє описувати поведінку, тобто алгоритми функціонування цифрових систем, а також проводити функціонально-структурний опис систем, має засоби для опису паралельних асинхронних процесів, регулярних структур і в той же час має всі ознаки мови програмування високого рівня - дозволяє створювати свої типи даних, має широкий набір арифметичних і логічних операцій.

Цифрова система на мові VHDL може бути описана на структурному і поведінковому рівнях.

Структурний опис – це опис системи у вигляді сукупності компонент (підсхем, елементів) і зв'язків між компонентами.

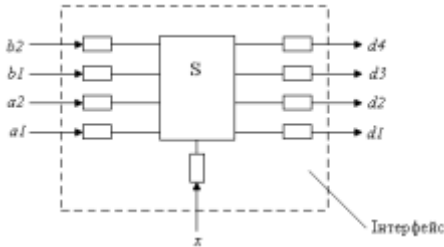


Рисунок 3.1 – Система S та її інтерфейс

S (рис. 3.1) подаються два дворозрядні числа $a = (a_2, a_1)$, $b = (b_2, b_1)$, де a_2, b_2 – старші розряди чисел a, b ; x – сигнал керування.

Якщо $x = 0$, то система S складає числа a, b і видає чотирирозрядний результат $d = (d_4, d_3, d_2, d_1) = (0, c_2, s_2, s_1)$, де c_2 – сигнал переносу.

Якщо $x = 1$, то система S перемножує числа a, b і видає чотирирозрядний результат $d = (d_4, d_3, d_2, d_1)$, де $d = a \cdot b$.

Виконання структурного опису системи S . На структурному рівні в систему входять: дворозрядний помножувач, суматор, пристрій керування, а також схема диз'юнктивного об'єднання вихідних сигналів (рис. 3.2).

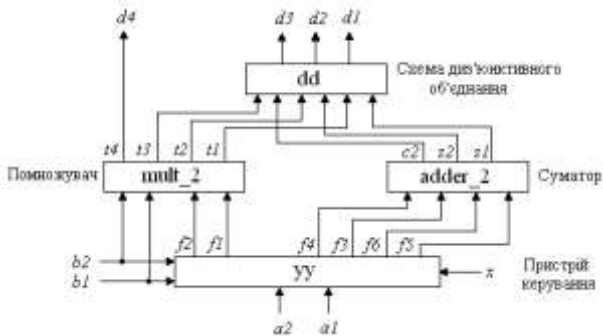


Рисунок 3.2 – Структура цифрової системи S

Алгоритм роботи пристрою керування:

якщо $x = 0$, то

$$(f_4, f_3) = (a_2, a_1),$$

$$(f_6, f_5) = (b_2, b_1),$$

$$(f_2, f_1) = (0, 0),$$

тобто числа a, b подаються на входи суматора;

якщо $x = 1$, то

$$(f_4, f_3) = (0, 0),$$

$$(f_6, f_5) = (0, 0),$$

$$(f_2, f_1) = (a_2, a_1),$$

тобто числа a, b подаються на входи помножувача.

Нехай опис системи матиме ім'я *VLSI_1*. Тоді ієрархія структурного опису системи *S* буде мати вид, представлений на рис. 3.3.

Структура блоку *mult_2* представлена на рис. 3.4.

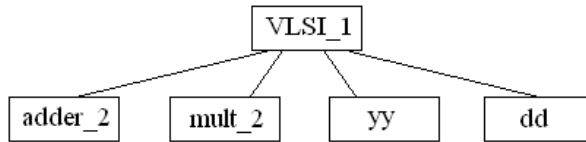


Рисунок 3.3 – Дерево ієрархії структурного опису системи *S*

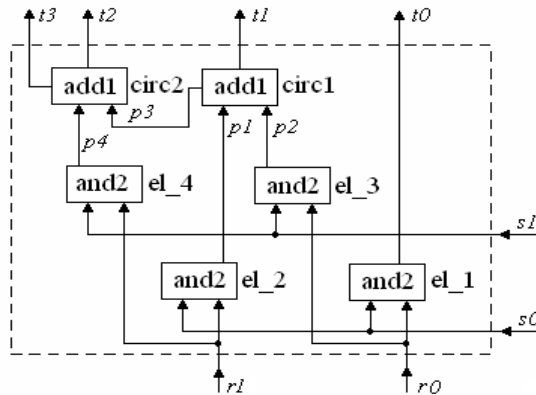


Рисунок 3.4 – Структура дворозрядного помножувача *mult_2*

Вхідні сигнали блоку *mult_2*:

$$r_1, r_0, s_1, s_0.$$

Сигнали $r1$, $r0$ інтерпретуються як дворозрядне ціле число $r = (r1, r0)$, сигнали $s1$, $s0$ – як дворозрядне число $s = (s1, s0)$.

Вихідні сигнали блоку `mult_2`:

$$t = (t3, t2, t1, t0) = (r1, r0) * (s1, s0).$$

У блоці `mult_2` елемент `add1` – однорозрядний півсуматор, функціонування якого описується таблицею істинності (табл. 3.1).

Таблиця 3.1 – Таблиця істинності півсуматора

$b1$	$b2$	$c1$	$s1$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Математичний запис булевих функцій, наведених у табл. 3.1:

$$s1 = b1 \oplus b2 = (b1 \wedge \overline{b2}) \vee (\overline{b1} \wedge b2),$$

$$c1 = b1 \wedge b2.$$

Запис на мові VHDL, що описує функціонування елемента `add1`:

```
s1 <= ((b1 and (not b2)) or (not b1) and b2);
```

```
c1 <= b1 and b2;
```

де **and**, **or**, **not** – логічні оператори.

Елемент `and2` – двовходовий кон'юнктор. Опис функції `and2` на мові VHDL:

```
y <= x1 and x2;
```

де $x1$, $x2$ – вхідні сигнали;

y – вихідний сигнал.

У дерево проекту схеми помножувача `mult_2` входять:

- елемент `and2`;
- підсхема `add1`.

У свою чергу, підсхема `add1` включає:

- `or2` – двовходовий диз'юнктор;
- `and2` – двовходовий кон'юнктор;
- `inv` – інвертор.

Елементи `and2`, `or2`, `inv` є *листями* проекту. Вони не мають складових частин і називаються *примітивами* проекту.

Примітив описується тільки на поведінковому рівні.

Об'єктами проекту для дворозрядного помножувача є mult_2, add1, and2.

Позначення кореня дерева (mult_2) є ім'ям проекту.

Кожен об'єкт проекту має 2 різні типи описів:

- опис об'єкта «в цілому» (*entity*);
- опис архітектури об'єкта (*architecture*).

Архітектура - це структура системи на функціональному рівні її опису.

Архітектурне тіло (*architecture*) визначає тіло об'єкта. У архітектурному тілі описуються функції або структура об'єкта проекту.

Опис об'єкта «в цілому» мовою VHDL носить назву «інтерфейс» об'єкта. Він складається з імені об'єкта і опису портів (входів і виходів) об'єкта.

Наприклад, опис об'єкта проекту and2 має вигляд:

entity and2 is	-- декларація імені об'єкта проекту;
port (x1, x2: in BIT;	-- декларація входних портів;
y2: out BIT);	-- декларація вихідного порту;
end and2;	
architecture functional of and2 is	-- декларація архітектури;
begin	
y <= x1 and x2;	-- опис функції об'єкта;
end functional;	

У даному прикладі BIT – це тип сигналу.

Як видно з опису об'єкта and2, з ключового слова **architecture** починається алгоритмічна частина опису (архітектурне тіло). Архітектурне тіло має своє унікальне ім'я functional, яке зв'язується (**is**) з інтерфейсом схеми.

Опис об'єкта add1 має вигляд:

```

entity add1 is
port (b1, b2: in BIT;
      c1, s1: out BIT);
end add1;

architecture struct_1 of add1 is
begin
    s1 <= ((b1 and (not b2)) or ((not b1) and b2));
    c1 <= b1 and b2;
end struct_1;
```

Опис об'єкта проекту mult_2 має вигляд:

```

entity mult_2 is
port (s1, s0, r1, r0: in BIT;
      t3, t2, t1, t0: out BIT);
end mult_2;

architecture structure of mult_2 is
component
add1 port (b1, b2: in BIT;
          c1, s1: out BIT);
end component;
signal p1, p2, p3, p4: BIT;
begin
    t0 <= r0 and s0;           -- елемент el_1;
    p1 <= r1 and s0;           -- елемент el_2;
    p2 <= r0 and s1;           -- елемент el_3;
    p4 <= r1 and s1;           -- елемент el_4;

    circ1: add1 port map (p1, p2, p3, t1);
    circ2: add1 port map (p3, p4, t3, t2);
end structure;

```

Пояснення. У описі архітектури проекту mult_2 оголошуються (декларуються) дві підсхеми (компоненти): circ1, circ2.

Після ключового слова **begin** надаються екземпляри описів, кожний з яких має унікальну мітку (circ1, circ2 – мітки), а також карту портів (**port map**).

Карта портів відображає зв'язок між входами, виходами описів компонента і екземплярами компонента.

У даному описі використовувалося поняття компонента (підсхеми) для add1, тоді як логічні елементи "І" схеми описані на функціональному рівні (без використання поняття компонента).

Можливість використання змішаних описів є важливою рисою особливості мови VHDL.

Опис дворозрядного суматора adder_2: дворозрядний суматор (рис. 3.5) складається з двох підсхем: add1 і add2, де add1 – одnorозрядний півсуматор, що вже розглядався, а add2 – одnorозрядний суматор, функціонування якого описується таблицею істинності (табл. 3.2).

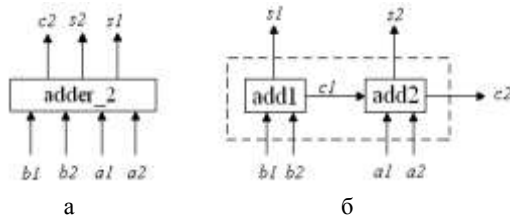
Умовне позначення дворозрядного суматора: **adder_2** (рис. 3.5).

Вихідні сигнали суматора формуються за формулою:

$$(a1, b1) + (a2, b2) = (c2, s2, s1).$$

Таблиця 3.2 – Таблиця істинності однорозрядного суматора

c1	a1	a2	c2	s2
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



а – умовне позначення; б - схема у вигляді каскадного з'єднання однорозрядного півсуматора add1 і однорозрядного суматора add2

Рисунок 3.5 - Дворозрядний суматор $(a1, b1) + (a2, b2) = (c2, s2, s1)$

У дерево проекту для підсхеми adder_2 входять підсхеми add1, add2. VHDL – опис підсхеми adder_2 має вигляд:

```

entity adder_2 is
  port (a1, b1, a2, b2: in BIT;
        c2, s2, s1: out BIT);
end adder_2;

architecture structure of adder_2 is
  component
    add1
    port (b1, b2: in BIT;
          c1, s1: out BIT);
  end component;

  component add2
    port (c1, a1, a2: in BIT;
          c2, s2: out BIT);
  end component;

  signal c1: BIT;
begin
    circ1: add1
    port map(b1, b2, c1, s1);

```

```

        circ2: add2
    port map(c1, a1, a2, c2, s2);
end structure;

```

Можна помітити, що в різні підсхеми входить add1, при цьому підсхема add1, що входить у суматор adder_2, є листом проекту, тому описана на поведінковому рівні.

Підсхема add1, що входить у помножувач mult_2, описана на структурному рівні.

VHDL – код для опису системи S має вигляд:

```

entity vlsi_1 is
port (a2, a1, b2, b1, x: in BIT;
        d4, d3, d2, d1: out BIT);
end vlsi_1;

architecture structure of vlsi_1 is
    component adder_2
    port (a1, b1, a2, b2: in BIT;
        c2, s2, s1: out BIT);
    end component;

    component mult_2
    port (s1, s0, r1, r0: in BIT;
        t3, t2, t1, t0: out BIT);
    end component;

    component dd
    port (x1, x2, x3, x4, x5, x6: in BIT;
        y1, y2, y3: out BIT);
    end component;

    component yy
    port (a2, a1, b2, b1, x: in BIT;
        f6, f5, f4, f3, f2, f1: out BIT);
    end component;

    signal f1, f2, f3, f4, f5, f6, t4, t3, t2, t1, c2, s2, s1: BIT;

    begin
        circ1: yy
        port map(a2, a1, b2, b1, x, f6, f5, f4, f3, f2, f1);
        circ2: mult_2
        port map(b2, b1, f2, f1, d4, t3, t2, t1);
        circ3: adder_2
        port map(f4, f3, f6, f5, c2, s2, s1);
        circ4: dd
        port map(s1, t1, s2, t2, c2, t3, d1, d2, d3);
    end structure;

```

3.2 Використання САПР «MAX+plus II» для розробки цифрових пристроїв на ПЛІС

САПР «MAX+plus II» є інтегрованим середовищем для розробки цифрових пристроїв на базі прогамованих логічних інтегрованих схем (ПЛІС) фірми «Альтера» і забезпечує виконання всіх етапів, необхідних для випуску готових виробів:

- створення проектів пристроїв;
- синтез структур і трасування внутрішніх зв'язків ПЛІС;
- підготовку даних для програмування або конфігурування ПЛІС (компіляцію);
- верифікацію проектів (моделювання і часовий аналіз);
- програмування або конфігурування ПЛІС.

Розробку функціонально-структурної схеми за її алгоритмічним описом називають високорівневим синтезом на відміну від логічного синтезу, коли за функціонально-структурним описом цифрової системи треба розробити логічну схему із заданих базисних логічних елементів.

Програму, яка здійснює синтез схеми на базі VHDL-опису, називають компілятором. Однак в системах моделювання VHDL-кодів під компіляцією також розуміється перетворення VHDL-коду в проміжну мову, яку безпосередньо використовують програми моделювання.

3.3 Процедура розробки проекту в САПР MAX+PLUS II

Процедура розробки проекту в САПР MAX+Plus II полягає у виконанні проектувальником нижченаведених поетапних дій.

Створення робочої папки для розміщення файлів проекту: у директорії MAXWORK необхідно створити робочу папку, наприклад, під ім'ям *vlsi_1*.

Створення директорії проекту: директорія створюється за допомогою завдання послідовності команд *File|Project|Name* і введенням імені проекту (наприклад, *vlsi_1*). При цьому вибирається створена робоча папка.

Створення текстового файлу: якщо текстовий файл, створений в будь-якому текстовому редакторі (наприклад, в редакторі Word),

вже існує, то необхідно його вміст записати в буфер пам'яті за допомогою введення команд *Правка|Виделить все|Ctrl+C*.

Далі слід активізувати текстовий редактор, задавши послідовність команд *Max+plusII|Text Editor*, а також перемістити інформацію з буфера пам'яті за допомогою натиснення поєднання клавіш *Ctrl+V*.

Якщо введений файл є програмою, написаною на мові VHDL, то його слід зберегти в директорії проекту з розширенням *.vhd*. При цьому задається послідовність команд *File|Save As* і вводиться ім'я файлу (наприклад, *vlsi_1.vhd*).

Виконання компіляції файлу: компіляцію виконують запуском додатку *Compiler*. За наявності помилок в програмі, їх слід усунути і виконати повторну компіляцію.

Створення Include-файлу і символу бібліотечного елемента: Include-файл створюється командами *File|Create Default Include File* і записується в бібліотеку користувача. Прочитати цей файл і уточнити назви входів і виходів можна за допомогою виклику додатку *Hierarchy Display*. При цьому відображаються всі модулі проекту і їх взаємозв'язки, а також всі типи файлів, сформовані в процесі обробки проекту. Include-файл відображається з розширенням *.inc*. Його активізація дозволяє проглянути вміст вказаного файлу.

Створення графічного файлу: для виклику графічного редактора потрібно в меню *Menager* вибрати *Max+PlusII|Grafic Editor*.

Графічному файлу із схемою необхідно командою *File|Save As* привласнити ім'я з розширенням *.gdf* (наприклад, *vlsi_1.gdf*).

Після того, як функціональні блоки введені, потрібно ввести символи входних і вихідних портів. Їх необхідно імпортувати з бібліотеки примітивів. Для цього необхідно двічі клацнути мишею по порожньому полю графічного редактора. Відкриється діалогове вікно, в якому в меню *Symbol Libraries* вказана бібліотека знаходиться за адресою *c:\program file\maxplusiiv10.2\max2lib\prim*. Після подвійного клацання за вказаною адресою в меню *Symbol File* з'явиться список логічних елементів. З вказаного списку необхідно вибрати примітиви портів, які зберігаються в бібліотеці під іменами *input* і *output*.

Далі необхідно привласнити імена всім портам.

Симуляція – це процес функціонального моделювання роботи схеми. Перед виконанням моделювання необхідно створити тестові вектори, тобто задати значення входних сигналів. Для цієї мети можна

використати редактор діаграм, який вибирається послідовністю команд *Max+PlusII|Waveform Editor*.

Коли вікно редактора відкриється, створюється файл (наприклад, з назвою *vlsi_1.scf*) послідовністю команд *File|Save As* і вказівкою назви файлу (наприклад, *vlsi_1.scf*) в рядку *File Name* діалогового вікна, що відкрилося.

Далі визначаються вхідні і вихідні сигнальні лінії схеми для процесу симуляції. Для цього використовуються сигнальні лінії, занесені в *SNF-файл (Simulator Netlist File)*, створений на етапі компіляції схеми. Необхідно відкрити список доступних в *SNF-файлі* сигнальних ліній за допомогою введення *Node|Enter Node from SNF*. Відкриється екран з двома вікнами: *Available Nodes & Groups* і *Selected Nodes & Groups*. Після натиснення *List* в першому вікні з'явиться список вхідних і вихідних ліній з *SNF-файлу*. Необхідно скопіювати список вхідних і вихідних ліній в друге вікно, тобто створити список вибраних сигнальних ліній. Після введення *OK* у вікні графічного редактора відобразяться вхідні і вихідні лінії.

Далі задається кінцевий час симуляції введенням *File|End Time* і інтервал часової сітки *Options|Grid Size*.

Для установки значень вхідних сигналів можна скористатися одним із способів: за допомогою вертикального репера встановити кліком з протяганням тривалість сигналу, потім увести його значення за допомогою кнопки символом "1" на лівій інструментальній панелі.

Для запуску пакету моделювання потрібно або ввести *File|Simulator*, або клацнути по кнопці симулятора на головній інструментальній панелі.

Якщо результати моделювання виявилися успішними, можна за допомогою виклику додатку *File|Timing Analyzer* відобразити таблицю "*Delay Matrix*", в якій записані затримки формування вихідних сигналів щодо вхідних сигналів.

Призначення ресурсів: для призначення ресурсів фізичних пристроїв і проглядання результатів розводки, зроблених компілятором, викликається порівневий планувальник *File|Floorplan Editor*. У вікні планувальника можна побачити тип мікросхеми, яка вибиралася в проєкті автоматично (при необхідності тип ПЛІС можна вибрати) і умовне графічне зображення вибраної ПЛІС з вказівкою під'єднуваних входів і виходів схеми.

3.4 Завдання

Розробіть проект під ім'ям *vlsi_1*. Порядок дій при розробці проекту описаний у підрозділі 3.3. Текстовий файл, який описує систему *S* (додаток А), надається викладачем.

При створенні графічного файлу використовуйте рис. 3.2.

При функціональному моделюванні роботи схеми, представленої графічним файлом, необхідно задати значення вхідних сигналів на різних часових інтервалах відповідно табл. 3.3.

Таблиця 3.3 – Значення коду (*x*, *b2*, *b1*, *a2*, *a1*)

№ вар.	Часові інтервали, нс			
	0...80	80...160	160...240	240...300
	<i>x</i> , <i>b2</i> , <i>b1</i> , <i>a2</i> , <i>a1</i>	<i>x</i> , <i>b2</i> , <i>b1</i> , <i>a2</i> , <i>a1</i>	<i>x</i> , <i>b2</i> , <i>b1</i> , <i>a2</i> , <i>a1</i>	<i>x</i> , <i>b2</i> , <i>b1</i> , <i>a2</i> , <i>a1</i>
1	0 0 0 1 0	0 1 1 1 0	1 1 1 0 1	1 1 0 0 1
2	0 1 1 1 1	0 1 0 0 0	1 0 1 0 0	1 0 1 0 1
3	0 0 0 0 1	0 1 0 1 0	1 1 0 0 1	1 0 0 1 0
4	0 1 0 0 1	0 0 1 1 1	1 0 1 1 1	1 1 0 1 0
5	0 0 0 1 0	0 1 0 1 0	1 1 0 0 1	1 0 1 1 1
6	0 1 0 1 0	0 0 1 1 1	1 0 0 1 0	1 0 0 0 1
7	0 0 1 0 0	0 1 0 1 0	1 1 1 1 0	1 0 1 1 1
8	0 1 0 0 1	0 1 0 1 0	1 0 0 1 0	1 0 1 0 1
9	0 0 1 1 0	0 1 0 0 1	1 1 0 1 1	1 0 0 0 1
10	0 1 0 0 1	0 0 1 1 1	1 0 1 0 0	1 1 0 1 0
11	0 0 1 1 0	0 1 0 0 1	1 1 0 0 1	1 0 0 0 1
12	0 1 0 1 0	0 0 0 1 0	1 0 0 1 0	1 1 1 0 0
13	0 0 1 0 1	0 1 0 0 1	1 1 0 0 0	1 0 0 0 1
14	0 1 0 1 0	0 1 1 1 0	1 0 0 1 0	1 1 0 1 0
15	0 0 0 0 1	0 1 1 0 0	1 1 0 0 1	1 0 0 1 0
16	0 1 0 1 0	0 0 0 0 1	1 0 0 1 0	1 0 0 0 1
17	0 0 0 1 0	0 1 0 1 0	1 1 0 1 0	1 0 1 1 0
18	0 1 1 1 1	0 1 0 0 1	1 0 1 0 1	1 1 1 1 1
19	0 0 1 0 1	0 1 1 0 1	1 1 0 1 0	1 0 1 0 0
20	0 1 1 0 1	0 1 0 0 0	1 0 0 0 1	1 0 1 0 1
21	0 0 0 1 0	0 1 0 0 1	1 1 0 1 0	1 0 1 1 1
22	0 1 1 1 0	0 0 0 1 0	1 0 1 1 1	1 1 0 1 0
23	0 0 0 1 0	0 1 0 0 0	1 1 1 0 0	1 0 0 0 1
24	0 1 0 0 1	0 1 0 0 1	1 0 0 1 0	1 0 0 0 1
25	0 0 1 0 1	0 1 1 1 0	1 1 0 0 0	1 0 0 0 1
26	0 1 0 1 0	0 0 1 1 1	1 0 1 0 0	1 1 0 1 0
27	0 0 0 0 1	0 1 0 1 0	1 1 1 0 1	1 0 0 0 1
28	0 1 1 0 0	0 1 0 0 0	1 1 1 0 1	1 1 0 0 1
29	0 0 0 1 0	0 1 1 1 1	1 1 1 0 1	1 0 1 0 1
30	0 1 1 1 0	0 1 0 1 0	1 0 1 1 0	1 0 0 1 1

3.5 Зміст звіту

Звіт повинен містити: мету роботи; зображення системи S та її інтерфейсу (рис. 3.1); структуру цифрової системи S (рис. 3.2); часові діаграми роботи системи S ; висновки.

3.6 Контрольні запитання

1. Що таке VHDL-опис, VHDL-код?
3. Чим структурний опис системи відрізняється від поведінкового?
4. Що таке проєкт, лист проєкту, примітив проєкту?
5. Що таке ієрархія проєкту?
6. Що таке високорівневий синтез?
7. Що таке логічний синтез?
8. Що таке архітектура об'єкта, архітектурне тіло (**architecture**)?
9. Що таке структура схеми, функція схеми, поведінка схеми?
10. Опишіть процедуру розробки проєкту в САПР MAX+PLUS II.
11. Чи можна на мові VHDL написати програму знаходження факторіалу натурального числа?
12. Чи правильне твердження: "VHDL має багато можливостей для моделювання аналогових схем"?
13. Чи правильне твердження: "Коментар в мові VHDL починається і закінчується двома дефісами"?
14. Скільки наведених нижче операторів відповідає виразу "заперечення кон'юнкції X , Y ", якщо відомо, що оператор **not** має найвищий пріоритет в порівнянні з іншими логічними операторами (тобто виконується першим)?
 - а) $Z \leftarrow \text{not } X \text{ and not } Y$;
 - б) $Z \leftarrow \text{not } (X \text{ and } Y)$;
 - в) $Z \leftarrow \text{not } X \text{ and } Y$;
15. Чи виконують приведені нижче оператори однакові функції?
 - а) $Z \leftarrow (A \text{ nand } B) \text{ nand } C$;
 - б) $Z \leftarrow A \text{ nand } (B \text{ nand } C)$;
 - в) $Z \leftarrow \text{not } X \text{ and } Y$;
16. Чи представляє тривходову NAND-комірку наведений нижче вираз?

$$A \text{ nand } B \text{ nand } C$$
17. Чим відрізняється процес симуляції від процесу компіляції?

4 ПРАКТИЧНЕ ЗАНЯТТЯ №4

"РЕАЛІЗАЦІЯ АВТОМАТНИХ VHDL-МОДЕЛЕЙ"

Мета роботи: ознайомлення з поняттям "підмножина мови VHDL, що синтезується"; розгляд методики опису кінцевого автомата мовою VHDL; виконання проекту пристрою, що реалізує заданий автомат з використанням VHDL-програми та аналіз його часових параметрів.

4.1 Теоретичні відомості

4.1.1 Підмножина мови VHDL, що синтезується

На базі VHDL-моделі синтезується схема, функції якої відповідають алгоритму VHDL-моделі.

Синтез здійснюється за допомогою спеціальних програм, які називають синтезаторами. Для кожного типу кристалів PLD або FPGA в синтезаторі є цільова бібліотека.

Основні кроки синтезу: створення проекту і установка опцій синтезу. Проект - це сукупність початкових VHDL-описів, необхідних пакетів, бібліотек, а також деякі внутрішні представлення, що вимагаються для роботи синтезатора.

Основними опціями синтезу є складність схеми «**Area**» (площа), швидкість «**Speed**», тип кодування даних (двійкове «**Binary**», унітарне «**One hot**», випадкове «**Random**», кодом Грея «**Gray**»), частота «**Clock Frequency**».

4.1.2 Моделювання VHDL-описів

Виконання VHDL-програми здійснюється за допомогою спеціальної програми - системи моделювання, яка включає:

- організацію проекту (визначення директорії проекту, розміщення в ній VHDL-кодів, необхідних пакетів, бібліотек тощо);
- компіляцію (**compile**) - перетворення VHDL-кодів у внутрішнє представлення, яке моделюється (компіляція не є синтезом логічної схеми);
- збирання (**link**) проекту;
- моделювання (**run**), тобто виконання VHDL-кодів, представлених у внутрішній формі;

- візуалізацію результатів.

Після того, як проектувальник переконується в коректності VHDL-моделі, виконується її схемна реалізація, тобто виконується синтез схеми. Зазвичай це інтегрована схема типу ПЛІС, або схема типу вентильної матриці.

При синтезі оператори мови VHDL замінюються компілятами.

Компілят - це підсхема, що реалізовує цілком певний оператор, наприклад, оператор складання.

При синтезі дані типу **bit** відповідають провідникам, типу **bit_vector** - шинам.

Проте перехід до відповідної логічної схеми здійснюється не для всієї мови VHDL, а тільки для деякої підмножини цієї мови, званої підмножиною, що синтезується.

Тільки для VHDL-моделі цифрової системи, описаної на підмножині мови VHDL, що синтезується, можна синтезувати схему.

Логічні оператори **and**, **or**, **xor**, **nand**, **nor**, **xnor**, **not** входять в підмножину, що синтезується.

Логічні оператори **and**, **or**, **xor** мають однакове старшинство і виконуються зліва направо у виразах.

Логічний оператор **not** має найвищий ранг і виконується раніше інших операторів:

$$Z \leq A \text{ and not } B \text{ or } C;$$

$$Z = \overline{AB} + C$$

В логічних операторах використовуються наступні типи даних: **boolean**; **bit**; **bit_vector**; **std_logic**; **std_logic_vector**; **std_ulogic**; **std_ulogic_vector**.

Тип **bit_vector** визначає масив бітів.

Приклади:

Bit_vector(0 to 3); --зростаючий діапазон

Bit_vector(7 downto 0); -- убуваючий діапазон

Приклад:

signal DataBus: bit_vector (7 downto 0);

1 0 0 1 0 1 0 1

7 6 5 4 3 2 1 0 № розряду

DataBus = "10010101";

DataBus(7)= "1";

.

DataBus(0)= "1";

4.1.3 Типи даних `std_logic`, `std_logic_vector`

Для опису вхідних і вихідних даних (кодів) синтезуємих схем використовуються тільки типи даних: **`std_logic`**, **`std_logic_vector`**.

Ці дані можуть приймати значення з багатозначної логіки:

- 'U' - не ініціалізоване;
- 'X' - невідоме значення (сильне джерело сигналу);
- '0' - логічний 0 (сильне джерело сигналу);
- '1' - логічна 1 (сильне джерело сигналу);
- 'Z' - високий імпеданс (ланцюг не підключений до джерела);
- 'W' - невідоме значення (слабке джерело сигналу);
- 'L' - логічний 0 (слабке джерело сигналу);
- 'H' - логічна 1 (слабке джерело сигналу);
- '-' - невизначене значення (байдужий стан).

Визначення типів **`std_logic`**, **`std_logic_vector`** міститься у пакеті `std_logic_1164`, там же міститься визначення підтипів, функцій на випадок багатозначної логіки. Щоб використати пакет `std_logic_1164`, необхідно задати:

```
Library IEEE;
Use IEEE.std_logic_1164.all;
```

Мова VHDL передбачає паралельну роботу процесів у часі. Тому сигналу одночасно можуть бути привласнені різні значення (у разі наявності у сигналу декількох джерел). Для визначення значення сигналу у цій ситуації використовується *функція перекриття*, яка визначає закон формування результуючого значення сигналу для всіх можливих комбінацій джерел сигналів. Принцип перекриття сигналів представлено на рис. 4.1.

Серед стандартних типів сигналів операція перекриття у VHDL визначена тільки для типів **`std_logic`** та **`std_logic_vector`**.

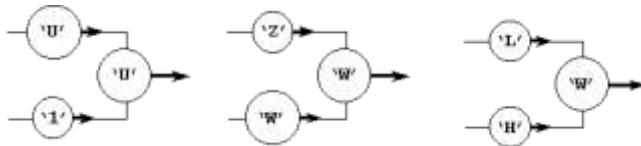


Рисунок 4.1 – Перекриття сигналів типу `std_logic`

Основне призначення типу **std_logic** - це дати можливість розробнику робити багатократні привласнення одному і тому ж сигналу.

Тип **std_logic** є підтипом типу **std_ulogic**.

Тип **std_logic** визначається в пакеті **std_logic_1164** так:

Subtype std_logic is resolved std_ulogic;

Тип **std_ulogic** є недозволяємим перераховувемим типом з множиною значень {'U','X','0','1','Z','W','L','H','-'}. Тобто сигнали цього типу не повинні мати багато джерел, їх визначення міститься в тому ж пакеті **std_logic_1164**.

4.1.4 VHDL - моделі кінцевих автоматів

ПЛІС, виконані за архітектурою FPGA, мають достатньо велику кількість тригерів, тому використання автоматних моделей дозволяє одержати достатньо швидкодіючу і в той же час наочну реалізацію пристрою при прийнятних витратах ресурсів.

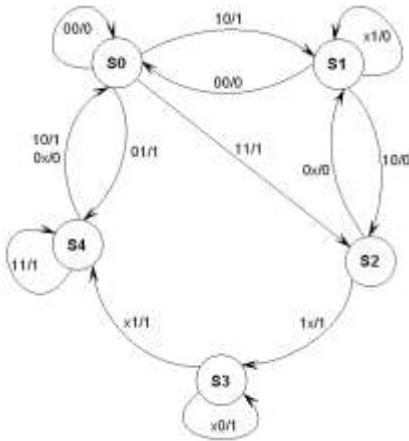
Нижче розглядається приклад проектування схеми на базі автомата Мілі.

VHDL – код реалізує поведінку кінцевого автомата, заданого графом (рис. 4.2, а). Архітектурний опис цього кінцевого автомата містить два внутрішні сигнали: *Pres_state* і *Next_state*. Сигнал *Pres_state* призначений для зберігання поточного стану автомата. Фізично він реалізується у формі тригерів (регістра відповідної розрядності). Сигнал *Next_state* використовується для визначення наступного стану – стану, який стане поточним у наступному такті. Значення цього сигналу визначається на базі значень вхідних сигналів і поточного стану автомата. Фізично цей сигнал реалізується у формі ліній зв'язку.

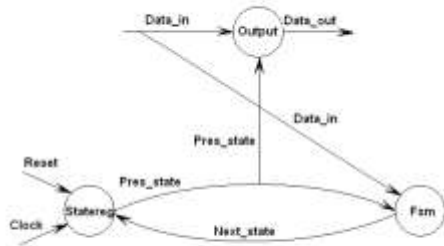
Поведінка автомата представляється у вигляді сукупності трьох процесів (рис. 4.2, б).

У процесі *Fsm* визначається наступний стан автомата. Фізична реалізація такого процесу, як правило, є комбінаційною схемою. В загальному випадку в тілі процесу можуть з'являтися оператори умовного переходу, в гілках яких визначаються значення для різних наборів сигналів. Тоді в схемі можуть з'явитися клямки.

Процес *Statereg* має список чутливості, в який входять сигнали *Reset* (обнулення) і *Clock* (синхросигнал). В процесі *Statereg* виконується перехід із стану у стан за відповідним фронтом сигналу *Clock*. За сигналом *Reset* = 0 автомат встановлюється в початковий стан *S0*.



а



б

а – граф автомата;

б - опис функціонування кінцевого автомата у вигляді сукупності процесів

Рисунок 4.2 – Кінцевий автомат Мілі з п'ятьма станами

Скидання автомата в початковий стан дозволяє забезпечити стабільну і безвідмовну роботу цифрового пристрою. Таким чином завжди забезпечується ініціалізація автомата в заздалегідь певному стані при першому тактовому імпульсі. Якщо скидання не передбачене, неможливо визначити, з якого стану почнеться функціонування, це може привести до збоїв в роботі всієї системи. Ймовірність виникнення такої ситуації збільшується при вмиканні живлення системи. Тому настійно рекомендується використовувати схеми скидання і початкової установки при проектуванні пристроїв на ПЛІС.

У процесі **Output** відповідно до поточного стану автомата визначається стан виходу **Data_out**.

4.2 Завдання

4.2.1 Розробка проекту Mealy

Виконайте VHDL-програму автомата Мілі з п'ятьма станами (додаток Б). Послідовності сигналів на шині **data_in** задайте у шістнадцятковій системі числення відповідно табл. 4.1. Часовий інтервал дії кожної з послідовностей задайте довільно.

риця часів передустановки і утримання сигналів) і **Registered Performance** (швидкодія регістрової логіки). При цьому піктограма **Delay Matrix** є активізованою. Після натиснення кнопки **Start** аналізатор часових параметрів обчислює затримки розповсюдження сигналів між вхідними і вихідними контактами поточного проекту.

Якщо шляхи розповсюдження сигналу мають різну довжину, то в комірці **Delay Matrix** з'являються два значення затримок, відповідні найдовшому і найкоротшому шляхам. Це означає, що в схемі є змагання сигналів. Коли в початковому проектному файлі джерело і приймач сигналу розділяються інформаційним входом *D*-тригера, затримка обчислюється через *Clock* (тактуючий) або *Preset* (встановлюючий) входи, а не через *D* (інформаційний) вхід.

При активізації **Setup/Hold Matrix** і кнопки **Start** визначте мінімально допустимі значення часів передустановки і утримання сигналів для інформаційних входів тригерів.

При активізації **Registered Performance** і кнопки **Start** визначте затримки в логіці між регістрами, мінімальний період і максимальну частоту тактового сигналу.

4.2.2.2 Робота в редакторі фізичного розміщення Floorplan Editor. Викличте редактор з основного меню **MAX+plus II**. В меню **Layout** представлені два варіанти зображення мікросхеми: **Device View** (показує всі контакти на корпусі мікросхеми та їх функції) і **LAB View** (вид логічних блоків, що показує логічні блоки *LAB* і логічні комірки *LC* усередині блоків, а також комірки вводу-виводу - *I/O cell*).

Виберіть команду **LAB View**. Прогляньте на екрані для кожного задіяного елементу вхідні і вихідні зв'язки. Для цього встановіть курсор на відповідну комірку і активізуйте її клацанням клавіші миші.

Використовуючи піктограми , розташовані зліва, виведіть на екран вхідні і вихідні зв'язки кожного елементу, розташованого в одній зайнятій комірці, а також зв'язки між елементами.

4.2.2.3 Отримання Delay Matrix. Для розрахунку **Delay Matrix** того вигляду, який Вам потрібен (бажано з більшою кількістю стовпців і рядків), на екрані редактора **Floorplan Editor** послідовно активізуйте вибрані комірки, викличте клацанням правої клавіші миші спливаюче меню, виберіть команду **Timing Analysis** та одну з вкладок:

Source (джерело сигналу, розташовується у рядку матриці), **Destination** (приймач сигналу, розташовується у стовпці матриці), **Cutoff** (вирізати з матриці). Потім запустіть **Timing Analyzer**.

4.3 Зміст звіту

Звіт повинен містити: мету роботи; граф кінцевого автомата Мілі з п'ятьма станами; опис функціонування кінцевого автомата у вигляді сукупності процесів; часову діаграму роботи автомата Мілі; матрицю затримок; мінімально допустимі значення часів передустановки і утримання сигналів для інформаційних входів тригерів; затримки в логіці між регістрами, мінімальний період і максимальну частоту тактового сигналу; висновки.

4.4 Контрольні запитання

- 1 Чим відрізняється підмножина мови VHDL, що синтезується, від множини мови VHDL?
- 2 Які параметри оптимального синтезу схеми можна змінити за допомогою послідовності команд **Assign/Global Project Logic Synthesis**?
- 3 На якому етапі розробки проекту створюються компіляти?
- 4 Чи можна стверджувати, що провіднику у схемі відповідає змінна типу **bit_vector**? Поясніть відповідь.
- 5 Навіщо необхідна передустановка сигналів для інформаційних входів тригерів?
- 6 Як сформулювати матрицю затримок бажаної форми?
- 7 Які процеси використовуються при описі поведінки автомата?
- 8 Які два варіанти зображення мікросхеми можна отримати в редакторі фізичного розміщення **Floorplan Editor**?
- 9 Як задати послідовності сигналів на шині **data_in** у різних системах числення?
- 10 Як отримати вхідні і вихідні зв'язки кожного елементу, розташованого в одній зайнятій комірці, а також зв'язки між елементами?
- 11 В яких випадках у комірках **Delay Matrix** з'являються два значення затримок?
- 12 Як визначити максимальну частоту тактового сигналу в проекті?

5 ПРАКТИЧНЕ ЗАНЯТТЯ №5

"ПРОЕКТУВАННЯ ЕЛЕКТРОННИХ СХЕМ НА ПЛІС"

Метою роботи є: ознайомлення з особливостями ПЛІС; розгляд принципів опису кінцевих автоматів; отримання навичок синтезу функціональних схем пристроїв при наявності автоматної моделі; розробка проекту пристрою з використанням САПР.

5.1 Теоретичні відомості

5.1.1 Особливості програмованих логічних інтегрованих схем

Програмовані логічні інтегровані схеми (ПЛІС) з'явилися як альтернатива програмованим логічним матрицям (ПЛМ). Від останніх ПЛІС відрізняються як за архітектурою, так і за технологією виготовлення.

ПЛМ є матрицею багатовходових (декілька десятків входів) логічних елементів з тригерами, в яких перемичками програмуються конститuentи одиниць диз'юнктивних нормальних форм функцій цих елементів. Спочатку перемички виконувалися у вигляді перепалюваних тонких провідників. Тепер перемички виконуються у вигляді МОН-транзисторів з плаваючим затвором, як в електрично перепрограмованому запам'ятовувальному пристрої (ППЗП), тобто ПЛМ виготовляються за технологією флеш-пам'яті. Великі ПЛМ (CPLD – Complex Programmable Logic Devices) відрізняються тільки тим, що декілька ПЛМ зібрані на одному кристалі і об'єднані програмованим полем зв'язків.

ПЛІС є матрицею маловходових (від двох до п'яти входів) логічних елементів, тригерів, відрізків ліній зв'язку, що сполучаються перемичками з польових транзисторів. Судячи з англійської назви - Field Programmable Gate Array (FPGA) - ПЛІС програмуються зміною рівня на затворах польових транзисторів, що використовуються для програмування. Ці транзистори підключені до входів тригерів одного довгого зсувового регістра, який заповнюється при програмуванні ПЛІС. Деякі з ділянок цього регістра можуть виконувати роль комірок ПЗП.

Прошивка зазвичай зберігається в ПЗП, що стоїть поряд з ПЛІС на платі і після приєднання живлення або за сигналом скидання вона автоматично переписується в програмуючий зсувовий регістр ПЛІС. Цей процес називається конфігурацією ПЛІС. Оскільки основу ПЛІС

складають тригери, що зберігають прошивку, то ПЛІС виготовляються за технологією мікросхем статичного ОЗП.

В порівнянні з CPLD, ПЛІС виграють у:

- необмеженій кількості перепрограмувань;
- логічній ємності;
- малому енергоспоживанні.

Як правило, ПЛІС мають на два - три порядки більшу ємність у кількості еквівалентних логічних вентилів, ніж CPLD, і, як статичне ОЗП, майже не споживають енергії за відсутності перемикань. Крім того, у ПЛІС на порядок вищий рівень надійності (нижча інтенсивність відмов), ніж у CPLD. До недоліків ПЛІС відносять необхідність зовнішньої ПЗП-прошивки, а також необхідність наявності генератора синхросерії. Але 8-выводове ПЗП займає на платі значно менше місця, ніж сама ПЛІС з багатьма сотнями виводів. Те ж стосується і генератора синхросерії.

Багато сумнівів у користувачів виникає із захистом проекту від копіювання. Дійсно, прошивка ПЛІС зберігається у зовнішньому ПЗП, вміст якого просто копіюється. Але змінити або розшифрувати прошивку, наприклад, для приховання авторства або відновлення схеми, практично неможливо, оскільки семантика бітів у файлі прошивки - секрет фірми, а необережна зміна її може вивести ПЛІС з ладу. Якщо потрібен захист, то завантаження програми виконують за допомогою зовнішньої CPLD, яка автоматично забезпечує захист проекту. У ПЛІС нових поколінь передбачається шифрування прошивки, наприклад, за допомогою вбудованого шифрувача DES із забезпеченням збереження ключа за допомогою батареї.

5.1.2 Опис алгоритму роботи кінцевого автомата

A finite state machine (FSM) - кінцевий автомат є моделлю поведінки, що складається з кінцевої кількості станів, переходів між цими станами і операцій. Кінцевий автомат - це абстрактна модель пристрою з примітивною оперативною пам'яттю.

Кінцеві автомати використовуються в багатьох областях, наприклад, в електротехніці, лінгвістиці, інформатиці, біології, математиці, логіці. У інформатиці кінцеві автомати широко використовуються в моделюванні поведінки додатків, проектуванні апаратних цифрових

систем, розробці програмного забезпечення, компіляторів, мережних протоколів.

Алгоритм роботи кінцевого автомата зручно описувати графом (рис. 5.1).

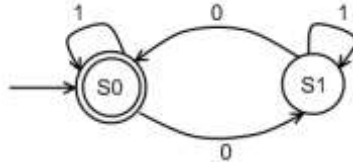


Рисунок 5.1 - Граф КА визначення парності кількості 0 в двійковому числі

Кінцевий автомат у кожен конкретний момент може знаходитися тільки в одному стані. Кожен тактовий імпульс може привести до переходу автомата в інший стан. Правила переходу визначаються комбінаційною схемою, званою логікою переходу. Наступний стан визначається як функція поточного стану. Стан виходу автомата у разі автомата Мілі визначається за допомогою логіки формування виходу.

Стан вводу (іноді розуміється як приймаючий стан) - стан, в якому машина успішно виконала свою процедуру. Він зазвичай відображається подвійним колом.

Початковий стан S_0 автомата, граф якого представлений на рис. 5.1, визначається як приймаючий стан. Цей автомат дасть правильний кінцевий стан, якщо двійкове число містить парне число нулів, або в числі немає нулів. Приклади рядків, прийнятих цим КА - епсілон (порожній рядок), 1, 11, 11..., 00, 010, 1010, 10110, і так далі.

5.2 Завдання

5.2.1 Постановка задачі

Синхронний автомат використовує два імпульси $Out1$ та $Out2$, що не перекриваються. Автомат приймає чотири стани: *Idle*, *Delay*, *Next*, *Done*. Граф автомата показаний на рис. 5.2.

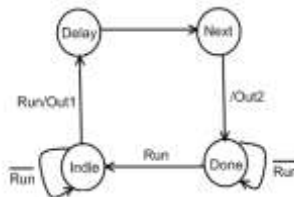


Рисунок 5.2 – Граф автомата з чотирма станами

Отже, для кодування станів потрібно два тригери. При використанні методу двійкового кодування можна записати систему рівнянь, що описує роботу автомата (символ " $\neg$ " відповідає операції НЕ):

$$\begin{aligned}
 Idle &= \neg S1 \cdot \neg S0; \\
 Delay &= \neg S1 \cdot S0; \\
 Next &= S1 \cdot S0; \\
 Done &= S1 \cdot \neg S0; \\
 S0 &:= (Idle \cdot Run) + Delay; \\
 S1 &:= (Done \cdot \neg Run) + Delay + Next; \\
 Out1 &:= Idle \cdot Run; \\
 Out2 &:= Next.
 \end{aligned} \tag{5.1}$$

У представленій системі рівнянь символ "=" позначає комбінаційну схему, відповідальну за перехід за станами, а символ ":=" позначає вихід тригера, необхідний для зберігання коду поточного стану автомата і вихідних сигналів.

Рівняння для вихідного сигналу *Out1* є функцією як стану, так і вхідного сигналу *Run*. Кінцевий автомат з таким видом стробування виходів називається автоматом Мілі.

5.2.2 Синтез функціональної схеми пристрою

5.2.2.1 На основі заданого графа побудуйте таблицю станів розробляемого автомата і таблицю призначених станів (з урахуванням перших чотирьох рівнянь системи (5.1)). Форми таблиць приведені у табл. 5.1 і табл. 5.2. У табл. 5.2 для позначення станів сигналу *Run* використайте тризначову логіку (0, 1, X).

Таблиця 5.1 – Форма таблиці станів автомата

Поточний стан	Наступний стан		Наступні вихідні стани		
			<i>Out1</i>		<i>Out2</i>
	<i>Run</i> = 0	<i>Run</i> = 1	<i>Run</i> = 0	<i>Run</i> = 1	<i>Run</i> = x
<i>Idle</i>					
<i>Delay</i>					
<i>Next</i>					
<i>Done</i>					

Таблиця 5.2 - Форма таблиці призначених станів автомата

Вхідний сигнал <i>Run</i>	Поточний стан		Наступний стан		Вихідні сигнали	
	$S1^n$	$S0^n$	$S1^{n+1}$	$S0^{n+1}$	<i>Out1</i>	<i>Out2</i>

5.2.2.2 За допомогою карт Карно мінімізуйте рівняння для змінних $S1^{n+1}$, $S0^{n+1}$.

5.2.3 Проектування і тестування проекту пристрою на основі моделі кінцевого автомата

5.2.3.1 Створення проекту. Створіть робочу папку для розміщення файлів проекту. Створіть директорію проекту.

Створіть графічний файл, для чого на основі приведених рівнянь (5.1) побудуйте схему з використанням графічного редактора MAX+plus II. У схемі передбачте три вхідні порти (*Run* для введення даних, синхровхід *Clk* і *Reset* для скидання тригерів у початковий стан), чотири вихідні порти: два для виведення однобітових вихідних даних *Out1* і *Out2*, два для контролю станів автомата (сигнали $S0$, $S1$). У схемі використовуйте *D*-тригери з інверсним входом скидання (чотири *D*-тригери відповідно до останніх чотирьох рівнянь системи логічних рівнянь, а також *D*-тригер, що прив'язує сигнал *Run* до синхроімпульсу).

5.2.3.2 Виконайте компіляцію графічного файлу.

5.2.3.3 Виконайте функціональне моделювання роботи розробленої схеми. На часовій діаграмі повинні бути відображеними три вхідні порти і чотири вихідні порти. Сигнал *Reset* на першому такті повинен бути одиничним, а на наступних тактах – нульовим. Задайте кінцевий час моделювання (приблизно 65 нс). Послідовність вхідних сигналів *Run* задайте відповідно до варіанту (табл. 5.3).

Таблиця 5.3 - Послідовність вхідних сигналів *Run*

№ вар.	Потактова послідовність вхідних сигналів <i>Run</i>	№ вар.	Потактова послідовність вхідних сигналів <i>Run</i>
1	0 0 1 1 0 0 1 0 0 1 0 1	16	1 0 0 0 0 0 1 1 1 1 0 1
2	1 1 0 0 1 1 1 1 0 1 0 1	17	1 1 1 0 0 0 1 1 0 1 0 1
3	0 1 0 1 0 0 1 0 1 1 0 1	18	1 0 0 0 0 0 1 0 1 0 0 1
4	1 0 0 1 1 0 0 0 1 1 0 0	19	1 0 1 1 1 0 1 1 1 1 0 0
5	1 0 0 1 0 0 1 1 1 1 0 1	20	1 0 1 0 1 0 1 1 1 0 0 1
6	0 1 1 1 0 0 1 1 0 1 1 0	21	0 0 1 0 1 1 1 1 0 0 0 1
7	1 1 0 1 0 0 0 1 1 0 0 1	22	1 0 1 0 1 0 1 1 1 1 0 1
8	1 0 1 1 0 0 0 1 0 0 0 1	23	1 1 1 0 0 0 1 0 1 0 0 1
9	0 0 0 1 1 0 1 1 1 1 1 0	24	1 0 1 1 0 0 1 1 1 1 0 1
10	1 1 1 0 0 0 0 1 0 1 0 1	25	1 0 0 0 0 0 1 0 0 1 0 1
11	1 0 1 1 1 0 1 0 1 1 0 0	26	1 1 0 1 0 1 1 1 0 1 0 1
12	0 0 1 1 0 0 1 0 0 1 0 1	27	1 1 0 1 0 0 0 1 1 1 0 1
13	1 1 1 0 0 1 0 1 0 0 0 1	28	1 0 1 1 0 0 1 1 1 1 0 1
14	1 0 1 0 1 0 1 0 1 1 0 1	29	1 1 1 0 0 0 1 1 0 0 0 1
15	1 1 0 1 0 0 1 0 0 1 1 0	30	1 0 1 0 1 0 1 0 0 1 0 1

5.2.4 Синтез функціональної схеми пристрою на основі мінімізованих рівнянь

5.2.4.1 Виконайте дії відповідно підпункту 5.2.3.1 з використанням отриманих у підпункті 5.2.2.2 мінімізованих рівнянь.

5.2.4.2 Виконайте дії відповідно до підпунктів 5.2.3.2 та 5.2.3.3.

5.2.4.3 Порівняйте результати моделювання схем на основі немінімізованих та мінімізованих рівнянь.

5.3 Зміст звіту

Звіт має містити: мету роботи; граф автомата з чотирма станами; рівняння системи (5.1), заповнені таблиці 5.1 та 5.2; схеми на основі немінімізованих та мінімізованих рівнянь; часові діаграми роботи системи за підпунктами 5.2.3.3 та 5.2.4.2; висновки за підпунктом 5.2.4.3.

5.4 Контрольні запитання

- 1 Які особливості мають ПЛІС?
- 2 Які особливості мають ПЛМ?
- 3 Навіщо на платі поруч з мікросхемою типу FPGA розташовують мікросхему ПЗП?

- 4 Які переваги мають ПЛІС порівняно з CPLD?
- 5 Зобразіть граф автомата, який визначає наявність у числі парної кількості одиниць або їх відсутність.
- 6 Скільки тригерів повинно бути у схемі автомата, яка його реалізовує, якщо автомат має 9 станів?
- 7 Які порти має програма з додатку Б?
- 8 Наведіть алфавіт тризнакової логіки.
- 9 У яких випадках автомат може знаходитись у приймаючому стані?
- 10 Навіщо у розробленій Вами схемі використовується вхід *Reset*?
- 11 Як за складністю відрізняються схеми, що побудовані на основі немінімізованих та мінімізованих рівнянь?
- 12 З яких етапів складається розробка проекту пристрою?

6 ПРАКТИЧНЕ ЗАНЯТТЯ №6

"РОЗРАХУНОК ПАРАМЕТРІВ І СИНТЕЗ ТОПОЛОГІЇ ПЛАНАРНОГО ДРЕЙФОВОГО *n-p-n* ТРАНЗИСТОРА"

Метою роботи є: вивчення структури і принципу роботи планарного біполярного транзистора; розгляд основних параметрів транзистора та методики їх розрахунку; програмна реалізація алгоритму синтезу топології відповідно до запропонованої методики.

6.1 Теоретичні відомості

6.1.1 Принцип дії і основні параметри транзистора

Характеристики та електричні параметри діодів і транзисторів напівпровідникових ІС визначаються електрофізичними параметрами матеріалів і геометричними розмірами напівпровідникових областей.

У прямому активному режимі роботи транзистора емітерний перехід зміщується у прямому напрямку, а колекторний - у оберненому. Неосновні носії заряду в базі (електрони в *n-p-n* транзисторі), які інжектуються емітером, переносяться від емітера до колектора в однорідній базі за допомогою дифузії, а в неоднорідній базі - за допомогою дифузії і дрейфу. У об'ємі бази відбувається рекомбінація неосновних і основних носіїв заряду. При тонкій базі ($W_B \ll L_n$) значна частина неосновних носіїв, які інжектуються

емітером, доходить до границі колекторного переходу, прискорюється полем цього переходу і переноситься в область колектора. При цьому створюється потік основних носіїв заряду.

Співвідношення між струмами транзистора у прямому активному режимі визначається коефіцієнтом передачі струму емітера α_N :

$$\alpha_N = \frac{I_K}{I_E} = \gamma \beta_N, \quad (6.1)$$

де γ - ефективність емітера або коефіцієнт інжекції емітера, рівний відношенню струму носіїв заряду, що інжектують крізь емітерний перехід, до повного струму емітера;

β_N - коефіцієнт переносу, що характеризує рекомбінаційні втрати неосновних носіїв заряду (електронів у *n-p-n* транзисторі) в базі.

У формулі (6.1) ефективність колектора $\gamma_E = I_{pE} / I_E$ дорівнює одиниці.

Коефіцієнт B_N характеризує величину підсилення струму бази:

$$B_N = \frac{I_K}{I_B}. \quad (6.2)$$

Коефіцієнти α_N і B_N пов'язані співвідношенням:

$$B_N = \frac{I_K}{I_B} = \frac{I_K}{I_E - I_K} = \frac{I_K / I_E}{1 - I_K / I_E} = \frac{\alpha_N}{1 - \alpha_N}. \quad (6.3)$$

Характеристичні довжини для розподілу донорів і акцепторів визначаються за формулами:

$$L_\partial = h_E / \ln \frac{N_{\partial E}(0)}{N_{\partial E}(h_E)}, \quad (6.4)$$

$$L_a = W_A / \ln \frac{N_{\partial E}(h_E)}{N_{\partial K}} = h_A / \ln \frac{N_{\partial E}(0)}{N_{\partial K}} = W_A / \ln \frac{N_{aA}(0)}{N_{\partial K}}, \quad (6.5)$$

де h_E, h_A - глибини залягання емітерного і колекторного переходів;

$W_B = h_B - h_E$ - товщина бази; $N_{\partial E}(0)$, $N_{\partial E}(h_E)$ - концентрації донорної домішки на поверхні емітерної області та поблизу емітерного переходу відповідно; $N_{aB}(0)$ - поверхнева концентрація акцепторної домішки у базі.

Початковий струм p - n переходу I_0 у загальному випадку містить у собі дифузійну, рекомбінаційну і генераційну складові:

$$I_0 = I_{0\partial u\phi} + I_{0pek} + I_{0gen}. \quad (6.6)$$

Електронний струм емітерного переходу:

$$\begin{aligned} I_{En} &= S_E q D_{nB} \frac{\partial n_p(x)}{\partial x} \Big|_{x=0} \cong \frac{S_E q D_{nB} \Delta n_p(0)}{W_B} = \\ &= S_E \frac{q D_{nB} n_{p0}}{W_B} (e^{U_{EB}/\varphi_T} - 1) = S_E \frac{q D_{nB} n_i^2}{N_{aB} W_B} (e^{U_{EB}/\varphi_T} - 1), \end{aligned} \quad (6.7)$$

де $S_E = R_E Z_E$ - площа емітера (добуток ширини R_E на довжину Z_E прямокутної емітерної області); $\varphi_T = \frac{kT}{q}$ - температурний потенціал; k - стала Больцмана; n_{p0} - рівноважна концентрація електронів у p -області.

Дірковий струм емітера при експоненціальному розподілі нерівноважної концентрації дірок в однорідно легованій n^+ -області визначається співвідношенням:

$$I_{Ep} = S_E \frac{q D_{pE} n_i^2}{N_{\partial E} L_{pE}} (e^{U_{EB}/\varphi_T} - 1). \quad (6.8)$$

У дрейфовому транзисторі з неоднорідно легованою базою закон розподілу концентрації акцепторної домішки близький до експоненціального:

$$N_{aB}(x) \cong N_{aB}(0) e^{-x/L_a} \quad \text{при } 0 \leq x \leq W_B,$$

де $L_a = W_B / \ln(N_{aB}(0) / N_{aB}(W_B))$.

У плавному колекторному переході $N_{aB}(W_B) \cong N_{\partial K}$.

Електронна складова струму емітера розраховується за співвідношенням:

$$I_{En} \cong S_E \frac{qD_{nB}n_i^2}{N_{aB}(0)W_B} \left(\frac{W_B}{L_a}\right) (e^{U_{EB}/\varphi_T} - 1) = S_E \frac{qD_{nB}n_i^2}{N_{\partial E}(h_E)L_a} (e^{U_{EB}/\varphi_T} - 1), \quad (6.9)$$

яке отримане за припущенням, що емітерний перехід при невеличких додатних зміщеннях можна вважати лінійним і прийняти $N_{aB}(0) \cong N_{\partial E}(h_E)$. Вираз перед дужками - це дифузійний компонент

$I_{E0\partial\phi}$ початкового струму емітерного переходу I_{E0} :

$$I_{E0\partial\phi} = S_E q \frac{D_{nB}n_i^2}{N_{\partial E}(h_E)L_a}. \quad (6.10)$$

З урахуванням лінійності емітерного переходу коефіцієнт інжекції емітера дрейфового транзистора дорівнює:

$$\gamma = \left(1 + \frac{I_{Ep}}{I_{En}}\right)^{-1} \cong 1 - \frac{L_{\partial}L_a}{\tau_{pE}D_{nB}} = 1 - \frac{L_{\partial}L_aD_{pE}}{L_{pE}^2D_{nB}}, \quad (6.11)$$

де $\tau_{pE} = \frac{L_{pE}^2}{D_{pE}}$ - час життя дірок у емітері.

Коефіцієнт переносу електронів у базі дрейфового транзистора з урахуванням виразу (6.5):

$$\beta \approx 1 - \frac{W_B L_a}{L_{nB}^2} = 1 - \left(\frac{W_B}{L_{nB}}\right)^2 \left(\ln \frac{N_{\partial E}(h_E)}{N_{\partial K}}\right)^{-1}. \quad (6.12)$$

У транзисторі з тонкою базою рекомбінаційна складова емітерного струму мала. Крім того, при створенні реальних транзисторів для зменшення швидкості поверхневої рекомбінації застосовують спеціальні технологічні операції. Таким чином, рекомбінаційна складова початкового струму емітерного переходу:

$$I_{E0pek} \cong 0. \quad (6.13)$$

Генераційний компонент повного теплового струму емітера описується виразом:

$$I_{E0_{ген}} = qS_E \frac{n_i d_E}{\sqrt{\tau_n \tau_p}}, \quad (6.14)$$

де d_E - ширина області просторового заряду емітерного переходу.

Емітерний перехід при прямих зміщеннях, як відзначалося вище, можна вважати плавним, з лінійною зміною концентрації домішки. У цьому випадку його ширина дорівнює:

$$d_E = \sqrt[3]{\frac{12\varepsilon_0(\varphi_{0E} + U_{EB})}{qa_{EB}}}, \quad (6.15)$$

де a_{EB} - градієнт концентрації домішок у емітерному переході.

У даному випадку:

$$a_{EB} = \frac{N_{\partial E}(h_E)}{L_a}. \quad (6.16)$$

З урахуванням (6.15), (6.16) і виразів $\sqrt{\tau_n} = L_n / \sqrt{D_n}$, $\sqrt{\tau_p} = L_p / \sqrt{D_p}$ початковий струм генерації в емітерному переході:

$$I_{E0_{ген}} = qS_E \frac{n_i \sqrt{D_{nB} D_{pE}}}{L_{nB} L_{pE}} \sqrt[3]{\frac{12\varepsilon\varepsilon_0(\varphi_{0E} + U_{EB})L_a}{qN_{\partial E}(h_E)}}. \quad (6.17)$$

Після підстановки (6.10), (6.13) і (6.17) у (6.6), одержимо:

$$I_{E0} = qS_E \left(\frac{D_{nB} n_i^2}{N_{\partial E}(h_E) L_a} + \frac{n_i \sqrt{D_{nB} D_{pE}}}{L_{nB} L_{pE}} \sqrt[3]{\frac{12\varepsilon\varepsilon_0(\varphi_{0E} + U_{EB})L_a}{qN_{\partial E}(h_E)}} \right), \quad (6.18)$$

де φ_{0E} - контактна різниця потенціалів емітерного переходу:

$$\varphi_{0E} = \varphi_T \ln\left(\frac{N_{\partial E}(h_E) N_{aB}(0)}{n_i^2}\right) = 2\varphi_T \ln\left(\frac{N_{\partial E}(h_E)}{n_i}\right). \quad (6.19)$$

У сучасних конструкціях напівпровідникових ІС біполярний планарний транзистор використовують не тільки в нормальному, але й у інверсному ввімкненні. В цьому випадку n - область кишень виконує роль емітера, а дифузійна p^+ - область - роль колектора.

Підсилювальні властивості планарного транзистора при інверсному ввімкненні недостатньо високі. Вони характеризуються інверсним коефіцієнтом підсилення:

$$B_I = (D_{nB} S_E K_X e^{\frac{W_B}{2L_a}}) / (D_{pK} S_B / L_{pK} + D_{nB} S_E M(K_X \text{cth}(K_X W_B) - \frac{1}{2L_a}) + D_{nB} (S_B - M S_E) (K_X \text{cth}(K_X h_B) - \frac{1}{2L_a})), \quad (6.20)$$

де M - число емітерів;

S_E, S_B - площі емітера і бази;

K_X - коефіцієнт, що визначається за формулою

$$K_X = \sqrt{\frac{1}{4L_a^2} + \frac{1}{L_{nB}^2}}.$$

Зворотний тепловий струм колектора в цьому випадку дорівнює

$$I_{K0} = q \left[\frac{\frac{S_E D_{nB} n_i^2 (K_X \text{cth}(K_X W_B) - \frac{1}{2L_a})}{N_{\partial K}} + \frac{S_B D_{pK} n_i^2}{N_{\partial K} L_{pK}} + \frac{S_B n_i \sqrt{D_{nB} D_{pK}}}{L_{nB} L_{pK}} \sqrt{\frac{12 \varepsilon \varepsilon_0 (\varphi_{0K} + U_{KB})}{q N_{\partial K} (\frac{1}{L_a} + \frac{1}{L_{\partial}})}} \right]. \quad (6.21)$$

де ε - діелектрична проникність напівпровідника;

ε_0 - діелектрична проникність вакууму;

φ_{0K} - контактна різниця потенціалів колекторного переходу:

$$\varphi_{0K} = \varphi_T \ln \left(\frac{N_{aB} (W_B) N_{\partial K}}{n_i^2} \right) = 2 \varphi_T \ln \left(\frac{N_{\partial K}}{n_i} \right). \quad (6.22)$$

Напівпровідниковий p - n перехід характеризується бар'єрною ємністю, що залежить від зворотної напруги зміщення U переходу:

$$C(U) = C_0(U) S = \frac{C_0(0) S}{(1 + U / \varphi_0)^m}, \quad (6.23)$$

де $C_0(U)$ - питома бар'єрна ємність p - n переходу при напрузі зміщення U ;

$C_0(0)$ - питома бар'єрна ємність p - n переходу при $U = 0$;

m - показник, обумовлений конструктивно-технологічними особливостями переходів ($m = 1/2$ для різкого переходу, $m = 1/3$ для плавного).

Питома ємність плавного p - n переходу визначається з виразу:

$$C_0(0) = \left(\frac{(\mathcal{E}_0)^2 q a}{12 \varphi_0} \right)^{1/3}. \quad (6.24)$$

Для переходів емітера і колектора маємо:

$$C_{EB0}(0) = \left(\frac{(\mathcal{E}_0)^2 q a_{EB}}{12 \varphi_{0E}} \right)^{1/3}, \quad (6.25)$$

$$C_{KB0}(0) = \left(\frac{(\mathcal{E}_0)^2 q a_{KB}}{12 \varphi_{0K}} \right)^{1/3}. \quad (6.26)$$

Після підстановки (6.25) і (6.26) відповідно в:

$$C_{EB}(U_{EB}) = S_E C_{EB0}(U_{EB}) = \frac{C_{EB0}(0) S_E}{(1 + U_{EB} / \varphi_{0E})^{1/3}}, \quad (6.27)$$

$$C_{KB}(U_{KB}) = S_K C_{KB0}(U_{KB}) = \frac{C_{KB0}(0) S_K}{(1 + U_{KB} / \varphi_{0K})^{1/3}} \quad (6.28)$$

і з огляду на формули:

$$a_{EB} = N_{\partial E}(h_E) \left(\frac{1}{L_\partial} + \frac{1}{L_a} \right),$$

$$a_{KB} = N_{\partial K} / L_a,$$

одержимо вирази для бар'єрних ємностей емітерного і колекторного переходів:

$$C_{EB}(U_{EB}) = S_E C_{EB0}(U_{EB}) = S_E \left(\frac{q N_{\partial E}(h_E) (\mathcal{E}_0)^2 \left(\frac{1}{L_\partial} + \frac{1}{L_a} \right)}{12 (\varphi_{0E} + U_{EB})} \right)^{1/3}, \quad (6.29)$$

$$C_{KB}(U_{KB}) = S_K C_{KB0}(U_{KB}) = S_B \left(\frac{q N_{\partial K} (\mathcal{E}_0)^2}{12 L_a (\varphi_{0K} + U_{KB})} \right)^{1/3}. \quad (6.30)$$

Для кремнієвих переходів з плавним (лінійним) розподілом домішок можна визначити напругу пробою як:

$$U_{\max} = 60 \left(\frac{3 \cdot 10^{20}}{a} \right)^{0,4}.$$

Звідси:

$$U_{EB \max} = 60 \left(\frac{3 \cdot 10^{20}}{N_{\partial E} (h_E) \left(\frac{1}{L_o} + \frac{1}{L_a} \right)} \right)^{0,4}, \quad (6.31)$$

$$U_{KB \max} = 60 \left(\frac{3 \cdot 10^{20} L_a}{N_{\partial K}} \right)^{0,4}. \quad (6.32)$$

Для максимальної напруги колектор-емітер маємо:

$$U_{KE \max} = \frac{2q N_{\partial K} W_B^3}{3 \mathcal{E}_0 L_a} - \varphi_{0K}. \quad (6.33)$$

Частота, на якій модуль коефіцієнта підсилення за струмом в схемі зі спільним емітером екстраполюється до одиниці, називається граничною частотою підсилення (граничною частотою, частотою відсікання). Вона пов'язана з фізичною структурою транзистора через час затримки τ носіїв, що пролітають від емітера до колектора:

$$f_T = \frac{1}{2\pi\tau}.$$

Час затримки τ являє собою суму чотирьох часів затримки, що характеризують послідовні фази прямування носіїв від емітера до колектора:

$$\tau = \tau_E + \tau_B + \tau_K + \tau_K',$$

де τ_E - час зарядки збідненого шару емітера;

$$\tau_B = \frac{L_a W_B}{D_{nB}} - \text{час прольоту носіїв крізь базу};$$

τ_K - час прольоту носіїв крізь збіднений шар колектора;

τ_K' - час зарядки бар'єрної ємності колектора.

З огляду на те, що $\tau_E, \tau_K, \tau_K' \ll \tau_B$, маємо:

$$f_T = \frac{1}{2\pi(\tau_E + \tau_B + \tau_K + \tau_K')} \cong \frac{1}{2\pi\tau_B},$$

або:

$$f_T = \frac{D_{nB}}{2\pi L_a W_B}. \quad (6.34)$$

Час розсмоктування заряду неосновних носіїв у транзисторі:

$$t_p = \frac{L_{nB}^2 L_{pK} h_B}{2D_{nB}(h_B L_{pK} / 2 + D_{pK} L_{nB}^2 / D_{nB})} \left(1 + \frac{2L_{pK}}{3h_B} \right). \quad (6.35)$$

6.1.2 Конструкції малопотужних планарних біполярних транзисторів

Типові конструкції малопотужних планарних біполярних транзисторів показані на рис. 6.1.

Вертикальна структура транзистора характеризується більш високими підсилювальними параметрами і швидкодією в порівнянні з горизонтальною. Проте реалізувати вертикальну конструкцію p - n - p транзистора разом із n - p - n транзистором технологічно складно.

Контактна n^+ -область потрібна для створення омичного контакту до ізольованої області n -типу провідності. Низькоомний схований n^+ -шар формується методом дифузії (або іонної імплантації з наступною розгонкою) миш'яку або сурми у вихідний підшарок для нарощування епітаксійної плівки і служить для зниження опору ізольованих n -областей. Знизити опір областей колектора і бази можна, збільшуючи площу контактів до цих областей. При цьому, проте, збільшуються розміри транзистора і, відповідно, бар'єрні ємності p - n переходів, що впливають на швидкодію.

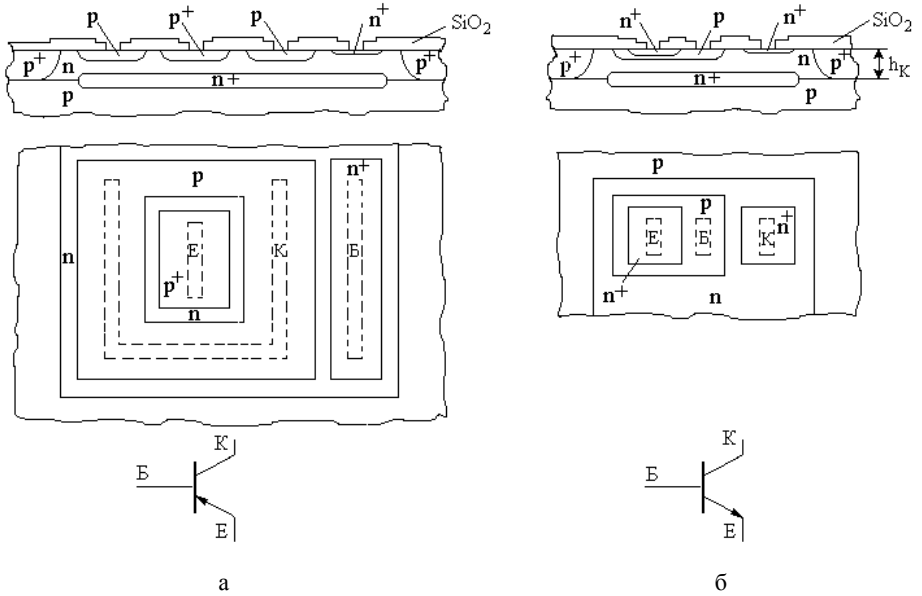


Рисунок 6.1 - Горизонтальна(а) та вертикальна (б) структури транзистора

6.1.3 Методика конструювання і розрахунку вертикального дрейфового планарного $n-p-n$ транзистора

При проектуванні та конструюванні елементів інтегрованих схем вихідними даними, як правило, бувають параметри відпрацьованої технології процесу виготовлення напівпровідникових ІМС та електрофізичні параметри матеріалів, що використовуються:

h_B - глибина залягання $p-n$ переходу база-колектор, см;

h_E - глибина залягання емітерного $p-n$ переходу, см;

h_K - товщина колекторної області (показана на рис. 6.1), см;

$N_{oE}(0)$ - концентрація донорної домішки в емітерній області на поверхні, см^{-3} ;

$N_{oE}(h_E)$ - концентрація донорної домішки в емітерній області поблизу емітерного переходу, см^{-3} ;

$N_{aB}(0)$ - поверхнева концентрація акцепторів у базі, см^{-3} ;

N_{oK} - концентрація донорної домішки у колекторі, см^{-3} ;

ρ_K - питомий об'ємний опір колекторної області, Ом·см;

$\rho_{\square БП}$, $\rho_{\square Ба}$ - питомий поверхневий опір пасивної й активної областей бази, Ом/□;

L_{pE} , D_{pE} - дифузійна довжина і коефіцієнт дифузії дірок у емітері, см і см²/с;

$L_{nБ}$, $D_{nБ}$ - дифузійна довжина і коефіцієнт дифузії електронів у базі, см і см²/с;

L_{pK} , D_{pK} - дифузійна довжина і коефіцієнт дифузії електронів у колекторі, см і см²/с;

n_i - концентрація носіїв зарядів у власному напівпровіднику, см⁻³;

ε - відносна діелектрична проникність напівпровідника.

У процесі конструювання при відомій структурі транзистора синтезується його топологія і розраховуються основні параметри:

I_{Kmax} - максимальний струм колектора, А;

B_N , B_I - нормальний і інверсний коефіцієнти передачі струму;

C_{KB} , C_{EB} - бар'єрні ємності p - n переходу, при заданих зворотних напругах U_{KB} і U_{EB} , В;

$U_{KE max}$, $U_{EB max}$, $U_{KB max}$ - максимальні зворотні напруги, В;

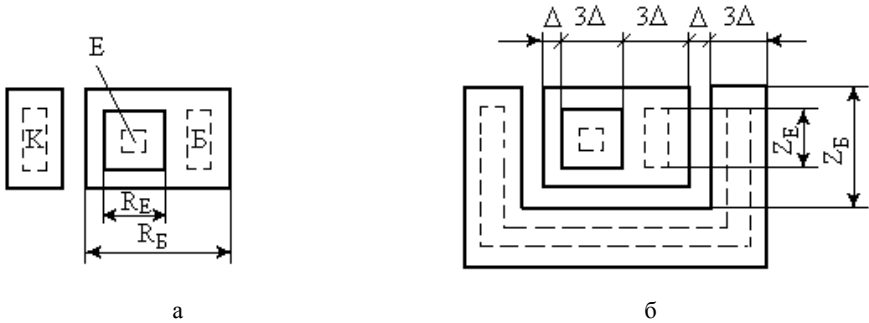
I_{K0} , I_{E0} - зворотні теплові струми колекторного і емітерного p - n переходів, А;

r_B , r_K - омичні опори областей бази і колектора, Ом;

f_T - гранична частота, Гц;

t_p - час розсмоктування заряду неосновних носіїв у транзисторі, с.

На практиці в рамках відпрацьованої технології для однієї і тієї ж структури розробляється декілька топологічних варіантів транзисторів (рис. 6.2), що відрізняються максимально допустимим струмом колектора, а відповідно, і розмірами. Тому, за основний заданий електричний параметр доцільно взяти саме максимальний струм колектора I_{Kmax} транзистора, синтезувати відповідну топологію транзистора, а інші електричні параметри розрахувати з урахуванням заданої структури та отриманої топології.



а - односмугова конструкція; б - конструкція з підвищеною контактною областю колектора

Рисунок 6.2 - Топологія *n-p-n* транзистора з одним базовим контактом

Якщо який-небудь із знайдених параметрів не задовольняє заданому, то треба в обґрунтованих випадках цілеспрямовано скорегувати технологію і повторити процес синтезу топології і розрахунку параметрів транзистора.

При автоматизованому синтезі топології і розрахунках параметрів транзистора доцільно уніфікувати конфігурації і характерні розміри Δ , що відповідають мінімальному розміру вікна на фотошаблоні (рис. 6.2). При цьому ширина емітерної області R_E для різноманітних варіантів залишається постійною, а змінюється довжина емітера Z_E і розміри бази R_B , Z_B .

Основні параметри дрейфового транзистора при малих і середніх рівнях струмів ($I_K = 0,1 \dots 50$ мА) визначаються за формулами, приведеними у пункті 6.1.1.

При малих розмірах транзистора варто враховувати, що бічна частина емітерного переходу має більш високу ємність C_{EB0_B} у порівнянні з торцевою C_{EB0_T} . Повну ємність можна в цьому випадку визначити за формулою:

$$C_{EB}(U_{EB}) = S_{E_T} C_{EB0_T} + S_{E_B} C_{EB0_B},$$

де S_{E_T} , S_{E_B} - площини торцевої і бічної частин емітера.

Методика синтезу конструкції транзистора заснована на розрахунку топологічних розмірів областей (розмірів на поверхні кристала) при заданому струмі колектора I_{Kmax} .

При синтезі топології з рішення трансцендентного рівняння (6.36) визначається мінімальний формат (відношення довжини до ширини) емітера Z_E / R_E , що забезпечує заданий розмір струму I_{Emax} з урахуванням ефекту витиснення струму до країв емітера:

$$\sqrt{0,9I_{Emax}G_E} = \sin\left(\sqrt{\frac{G_{Emax}}{0,9}}(1 - 0,9I_{Emax}G_E)\right); \quad (6.36)$$

де
$$G_E = \frac{R_E}{8Z_E\phi_TB_N} \cdot \rho_{Ba};$$

$$I_{Emax} = B_N I_{Kmax} / (1 + B_N).$$

Якщо розрахунок дає розмір відношення $Z_E / R_E < 1$, то приймається $Z_E = R_E$.

При більшому розмірі відношення Z_E / R_E доцільно довгу емітерну смугу розділити на декілька коротких емітерів.

В аналізованій методиці число емітерів вибирається з нерівності

$$1 \leq Z_E / R_E \leq 3(M+1),$$

де M - кількість емітерів.

Значення омичних опорів областей транзистора типових структурних конструкцій можна оцінити за формулами:

$$r_b = [(1-K_K)Z_E/(\Delta \cdot \rho_{Ba}) + 4K_K Z_E/(\Delta \cdot \rho_{B\Pi})]^{-1} \text{ при } M=1;$$

$$r_b = [((1-K_K)Z_E/(\Delta \cdot \rho_{Ba})) + (4Z_E/(\Delta \cdot \rho_{Ba}(M-1))) + ((1+K_K)Z_E/(\Delta \cdot \rho_{B\Pi})) + 4(M-1)Z_E/(\Delta \cdot \rho_{B\Pi})]^{-1} \text{ при } M>1;$$

$$r_K = \rho_K (h_K - h_B) \left[\frac{1}{S_E M} + \frac{1}{12\Delta^2 + 6\Delta \cdot Z_B + 3\Delta \cdot R_B} \right]^{-1},$$

де $K_K = 0$ для конструкції з одним базовим контактом (рис. 6.2);

$K_K = 1$ для багатобазової конструкції;

Δ - мінімальний розмір вікна на фотошаблоні.

Введення формального параметра K_K полегшує побудову програми автоматизованого проектування транзисторів.

Струм колектора транзистора визначається виразом:

$$I_K = \alpha_N S_E \frac{q D_{nB} n_i^2}{N_{\partial E} (h_E) L_a} \exp\left(\frac{U_{BE}}{\varphi_T} - 1\right). \quad (6.37)$$

6.1.4 Алгоритм розрахунку параметрів і синтезу топології біполярного *n-p-n* транзистора

Позначення вихідних даних перелічені в підрозділі 6.1.3, формули розрахунку параметрів і топологічних розмірів наведені в підрозділах 6.1.1, 6.1.3.

Алгоритм складається з нижченаведених пунктів:

- 1) розрахунок параметрів: $L_a, L_{\partial}, K_X, B_N, U_{KBmax}, U_{EBmax}, U_{KEmax}, f_T$.
- 2) синтез топології - визначення розмірів: $M, Z_E, R_B, Z_B, S_E, S_B, R_E$.
- 3) розрахунок параметрів: $B_I, C_{KB}(U_{KB}), C_{EB}(U_{EB}), I_{E0}, I_{K0}, t_p, r_B, r_K, I_{Kmax}$. (струм I_{Kmax} розраховується за формулою (6.37) при підстановці у вираз напруги U_{BE} , що падає на відкритому емітерному переході);
- 4) перехід до пункту 2, якщо розрахований максимальний колекторний струм перевищує заданий I_{Kmax} . При цьому кількість емітерів збільшується.

6.2 Завдання

6.2.1 Розрахуйте параметри транзистора і синтезуйте його топологію відповідно до приведеного алгоритму. Варіанти завдань приведені у додатку В.

6.2.2 Розрахуйте і побудуйте наведені залежності (за вибором викладача):

- а) коефіцієнта підсилення B_N від концентрації донорів у колекторі;
- б) нормального коефіцієнта передачі від товщини бази;
- в) зворотного теплового струму від температури;
- г) ємності емітерного (колекторного) переходу від прикладеної до нього зворотної напруги.

6.3 Зміст звіту

Звіт має містити: структуру транзистора, що проектується; розгорнутий алгоритм розрахунку параметрів і топологічних розмірів транзистора; результати розрахунків; топологію розробленого транзистора з дотриманням масштабних співвідношень; розраховані залежності.

6.4 Контрольні запитання

- 1 Як працює біполярний транзистор в нормальному активному режимі?
- 2 Яким коефіцієнтом визначається співвідношення між емітерним і колекторним струмами транзистора?
- 3 Який параметр називається ефективністю емітера?
- 4 Які складові містить зворотний струм p - n переходу I_0 ?
- 5 Яким співвідношенням пов'язані коефіцієнти α_N і B_N ?
- 6 Чи залежить від площі емітера електронний і дірковий струми емітерного переходу?
- 7 Яку базу необхідно створювати у транзисторі для зменшення рекомбінаційної складової базового струму?
- 8 Які особливості роботи та підсилювальні властивості має біполярний планарний транзистор у інверсному ввімкненні?
- 9 Від чого залежить бар'єрна ємність p - n переходу?
- 10 Якими параметрами визначається швидкодія транзистора при перемиканнях?
- 11 Чим горизонтальна структура транзистора відрізняється від вертикальної?
- 12 На чому заснована методика синтезу конструкції транзистора?
- 13 З яких пунктів складається алгоритм розрахунку параметрів і синтезу топології біполярного n - p - p транзистора?
- 14 Який прийом використовується при синтезі топології транзистора, якщо отриманий велике значення відношення Z_E / R_E ?

7 ПРАКТИЧНЕ ЗАНЯТТЯ №7

"РОЗРАХУНОК ПАРАМЕТРІВ І СИНТЕЗ ТОПОЛОГІЇ ЛЕ АБО-НЕ НА МОН-ТРАНЗИСТОРАХ"

Метою роботи є: ознайомлення з моделлю МОН-транзистора; розгляд особливостей проектування топології мікросхем на МДН-транзисторах; визначення параметрів МОН-структури; розрахунок електричних, часових, топологічних параметрів логічного елемента АБО-НЕ на МОН-транзисторах.

7.1 Теоретичні відомості

7.1.1 МДН-транзистори в інтегрованих мікросхемах

У даний час значне поширення одержали мікросхеми на МДН-транзисторах, що мають структуру метал-діелектрик-напівпровідник.

Мікросхеми на МДН-транзисторах мають просту конструкцію, вони технологічні, мають високий відсоток виходу придатних схем і не потребують додаткової ізоляції елементів на кристалі. МДН ІС характеризуються високою компактністю розміщення елементів, малою потужністю розсіювання, високою завадостійкістю, низькою вартістю.

На основі МДН-транзисторів випускається більш 50% вироблених у світі мікросхем. Цифрові ВІС на МДН-структурах досягли в даний час найвищого ступеня інтеграції, тому що на площі підшарку, яка займається одним біполярним транзистором, можна розмістити 100...150 МДН-структур.

У основі виготовлення мікросхем на МДН-транзисторах так само, як і біполярних мікросхем, лежить планарна технологія. Проте, число технологічних операцій, необхідних для створення мікросхем на МДН-транзисторах, значно менше, і тому процес їх виготовлення є менш критичним і складним.

У приповерхневому шарі напівпровідника методами локальної дифузії або йонного легування створюються стік і витік, відділені один від одного та від іншої частини кристала *p-n* переходами. За допомогою поперечного електричного поля, зосередженого в просторі між напівпровідником і затвором, створюється провідний канал. Такий транзистор називають структурою з індукованим каналом. У МДН-транзисторах з вбудованим каналом провідний канал створюється технологічними методами.

У залежності від типу носіїв заряду, що підтримують струм у каналі, розрізняють МДН-транзистори з n -каналом і p -каналом.

Витік n -канального МДН-транзистора (рис. 7.1) частіше з'єднується з кристалом мікросхеми (підшарком) і має нульовий потенціал.

Одним з основних етапів проектування мікросхем на МДН-структурах є вибір фізичної структури МДН-транзистора: питомого опору матеріалу підшарку, товщини діелектрика, матеріалу затвора.

При виготовленні мікросхем зазвичай використовують кремній із питомим опором $1 \dots 10 \text{ Ом} \cdot \text{см}$. Вибір підшарків з орієнтацією (100) кращий у порівнянні з орієнтацією (111), тому що густина поверхневих станів на таких підшарках майже на порядок менше. Частіше усього як підзатворний діелектрик використовується діоксид кремнію SiO_2 . Товщина підзатворного діелектрика може складати від $0,02$ до $0,1 \text{ мкм}$ і менше, а товщина захисного окислу 1 мкм .

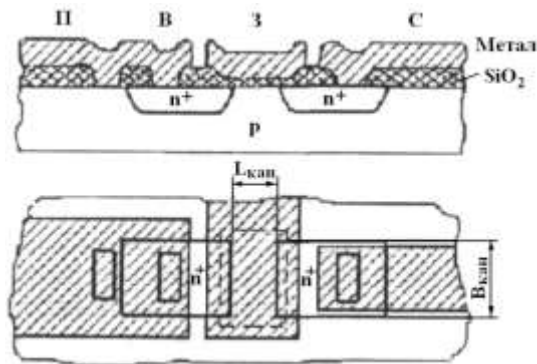


Рисунок 7.1 - Конструкція n -МОП-транзистора

Матеріалами затвора і міжз'єднань бувають алюміній, полікремній, тугоплавкі метали і їхні силіциди.

Напруга затвора, при якій виникає індукований канал, називається пороговою напругою $U_{\text{пор}}$.

У реальних МДН-транзисторах при визначенні порогової напруги необхідно враховувати заряд у діелектричній плівці і заряд поверхневих станів напівпровідника.

Вмонтований позитивний заряд діоксиду кремнію у підшарку утворюється в процесі виробництва транзисторів. Він зумовлений сторонніми позитивними домішковими йонами, наприклад, водню (H^+), натрію (Na^+).

Заряд поверхневих станів напівпровідника виникає тому, що поверхня напівпровідника являє собою порушення періодичності кристалічної ґратки. У кремнії, наприклад, на поверхні кристала знаходяться три сусідніх атоми замість чотирьох. Відсутній один електронний зв'язок. У результаті цього в забороненій зоні напівпровідника виникають додаткові акцепторні рівні.

Локальні енергетичні рівні, обумовлені порушенням періодичності кристала на поверхні або домішками на поверхні, називаються *поверхневими рівнями*.

Позначимо густину сумарного позитивного заряду Q_{ss} . Вона складається з густини заряду в діелектричній плівці і густини заряду поверхневих станів.

Цей заряд компенсується негативним зарядом електронів, які накопичуються в поверхневому шарі напівпровідника, що сприяє утворенню каналу n -типу провідності. На Q_{ss} впливає кристалографічна орієнтація поверхні кремнію і технологія виготовлення окисного шару (числове значення Q_{ss} змінюється зазвичай в межах від 10^{-9} до $5 \cdot 10^{-8}$ Кл/см²).

Накопиченню електронів біля поверхні напівпровідника, які полегшують утворення n -каналу, сприяє і те, що робота виходу з металу ψ_M менше роботи виходу з напівпровідника ψ_S у конструкціях МОН-транзисторів з алюмінієвими затворами. Якщо $\psi_M < \psi_S$, поверхня напівпровідника заряджатиметься негативно електронами, які приходять з напівпровідника. Цей заряд буде накопичуватися доти, поки в окисі не виникне електричне поле. У цьому разі між поверхнями металу і напівпровідника виникне контактна різниця потенціалів.

При визначенні контактної різниці потенціалів $q\psi_{MS}$ можна скористатися табл. 7.1.

Таблиця 7.1 - Контактна різниця потенціалів у вольтах

Тип затвору	Канал n -типу	Канал p -типу
Al-затвор	$-0,60 - \varphi_B $	$-0,60 + \varphi_B $
Si-затвор	$0,55 + \varphi_B $	$-0,55 - \varphi_B $

Примітка: $q\varphi_B = E_I - E_F$,

де E_i - енергія, що відповідає середині ширини забороненої зони; E_F - енергія Фермі.

Потенціал φ_B , що визначає положення рівня Фермі щодо середини забороненої зони в напівпровіднику p -типу електропровідності з концентрацією акцепторної домішки N_A , визначається:

$$\varphi_B = \varphi_T \cdot \ln \frac{N_A}{n_i},$$

де φ_T - температурний потенціал;

n_i - власна концентрація електронів у напівпровіднику.

Отже, для МДН-транзистора з каналом n -типу й алюмінієвим затвором наявність позитивного заряду Q_{SS} і контактної різниці потенціалів сприяє утворенню індукованого каналу. У результаті навіть при термодинамічній рівновазі (без прикладення будь-яких потенціалів до виводів транзистора) у напівпровіднику спостерігається невеличкий початковий вигин зон і утворення запираючого шару (рис. 7.2,а).

При прикладенні позитивної напруги до затвору спостерігається ще більший вигин зон, що приводить до ще більшого збіднення приповерхневої області напівпровідника основними носіями (дірками). Товщина збідненого шару зростає при зростанні напруги на затворі. При ще більших позитивних напругах (рис. 7.2, б) зони вигинаються униз настільки сильно, що поблизу поверхні відбувається перетинання рівня Фермі з рівнем E_i (рівень Фермі в напівпровіднику залишається постійним, тому що в ідеальній МДН-структурі струм затвора нульовий). У цьому випадку концентрація неосновних носіїв (електронів) у поверхні перевищує концентрацію основних носіїв (дірок). Такий режим називається інверсією. При цьому утворюється канал з електропровідністю n -типу. У момент утворення каналу товщина збідненої області досягає максимального розміру. Товщина провідного каналу складає усього 1...2 нм, що в сотні разів менше товщини збідненого шару.

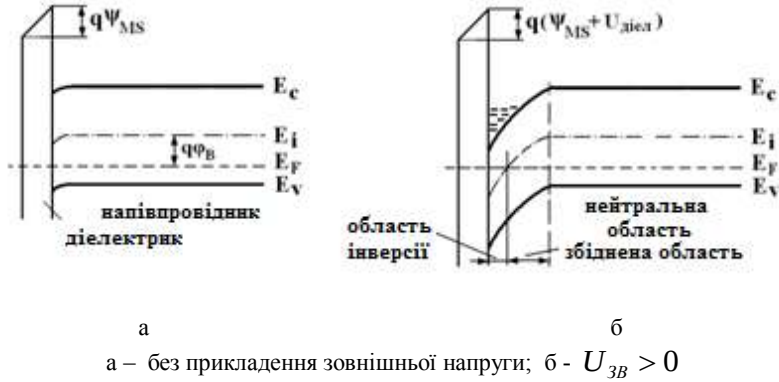


Рисунок 7.2 - Утворення запираючого шару в області контакту метал – діелектрик - напівпровідник при $\psi_M < \psi_S$

Можна довести, що інверсія настає при потенціалі на поверхні напівпровідника, рівному $2\varphi_B$.

Якщо не враховувати початковий вигин зон, що має місце і при відсутності напруги на затворі, можна сказати, що прикладена до затвору напруга падає на діелектрику і напівпровіднику. Отже:

$$U_{\text{ПОР}} = 2\varphi_B + U_{\text{ДІЕЛ}},$$

де $U_{\text{ДІЕЛ}}$ - падіння напруги на діелектрику, рівне

$$U_{\text{ДІЕЛ}} = Q_A / C_0,$$

де Q_A - густина заряду збідненої області:

$$Q_A = q \cdot N_A \cdot Z.$$

Товщина збідненого шару Z обчислюється за формулою:

$$Z = \sqrt{\frac{2\varepsilon_0(2\varphi_B)}{qN_A}}.$$

З урахуванням початкового вигину зон:

$$U_{\text{ПОР}} = 2\varphi_B + \frac{Q_A}{C_0} - \frac{Q_{SS}}{C_0} + \psi_{MS}. \quad (7.1)$$

При визначенні $U_{\text{ПОР}}$ необхідно враховувати, що ψ_{MS} для МДН-транзисторів з алюмінієвим затвором від'ємна.

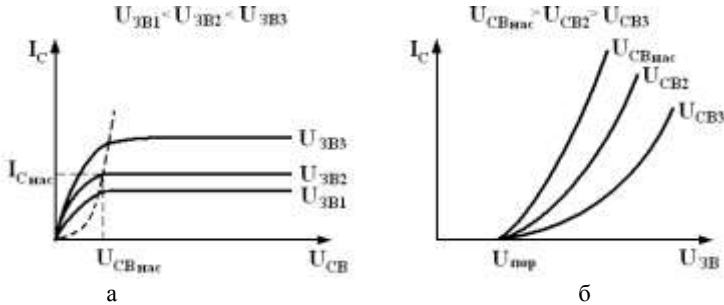
Питома ємність між затвором і каналом дорівнює:

$$C_0 = \varepsilon_D \varepsilon_0 / d ,$$

де d - товщина підзатворного діелектрика;

ε_D - його діелектрична проникність.

Статичні характеристики МДН-транзистора з індукованим n -каналом показані на рис. 7.3.



а – вихідні характеристики; б – передавальні характеристики

Рисунок 7.3 - Статичні характеристики МДН-транзистора з індукованим n -каналом

Крута ділянка вихідної характеристики транзистора апроксимується формулою:

$$I_C = b(U_{CBнас} \cdot U_{CB} - 0,5 \cdot U_{CB}^2),$$

де $U_{CBнас} = U_{ZB} - U_{ПОР}$.

Струм насичення транзистора дорівнює:

$$I_{Cнас} = \frac{b}{2} \cdot U_{CBнас}^2 ,$$

де b - питома крутість МДН-транзистора (один із його основних параметрів):

$$b = \mu_n C_0 \frac{B_{КАН}}{L_{КАН}} ,$$

де μ_n - приповерхнева рухливість носіїв (вона зазвичай у 2-3 рази менше об'ємної).

Логічний елемент АБО-НЕ на МДН-транзисторах показаний на рис. 7.4.

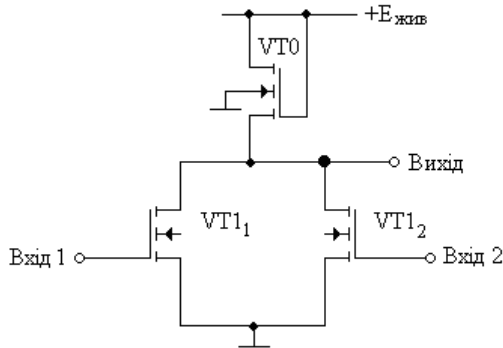


Рисунок 7.4 - Схема електрична логічного елемента АБО-НЕ

Схема працює так: при низькому потенціалі на всіх затворах $VT1$ ці транзистори закриті, і струми стоків $VT1$ рівні нулю. В схемі не тече струм, тому на виході схеми встановлюється високий потенціал U^1 .

При високому потенціалі на затворі хоча б одного транзистора $VT1$ цей транзистор відкривається, у схемі з'являється струм, напруга на виході схеми стає малою і рівною падінню напруги на відкритому транзисторі $VT1$, що називається залишковою напругою $U_{3\text{Л}}$.

Наближені аналітичні формули для розрахунку основних параметрів логічного елемента АБО-НЕ приведені в табл. 7.2. У таблиці C_{Π} - сумарна паразитна ємність схеми; M_0 - кількість входів елемента.

Таблиця 7.2 - Формули для розрахунку параметрів ЛЕ АБО-НЕ

Параметри	Формули
Логічні рівні	$U^1 = E_{\text{жив}} - U_{\text{пор}}$ $U^0 = 0,5(b_0/b_1) \cdot ((E_{\text{жив}} - U_{\text{пор}})^2 / M_0(E_{\text{жив}} - 2U_{\text{пор}}))$
Завадостійкість	$U_3^+ = U_{\text{пор}} - U_{\text{зал}}; \quad U_3^- = E_{\text{жив}} - 2U_{\text{пор}}$
Потужність споживання	$P = 0,25b_0E_{\text{жив}}(E_{\text{жив}} - U_{\text{пор}})^2$
Середня затримка перемикання	$t_z = \frac{C_{\Pi}}{(E_{\text{жив}} - 2U_{\text{пор}})} \cdot \left[\frac{1}{b_1M_0} + \frac{U_{\text{пор}}}{b_0(E_{\text{жив}} - U_{\text{пор}})} \right]$

7.1.2 Проектування топології логічного елемента АБО-НЕ

Проектування топології МДН-мікросхем є більш простим порівняно з проектуванням мікросхем на біполярних транзисторах. Це зумовлено простотою виготовлення МДН-мікросхем, особливо схем з каналами одного типу провідності (немає епітаксійних і схованих шарів, а також відсутність спеціально утворюваної електричної ізоляції елементів). Крім того, схемотехнічна побудова МДН-мікросхем більш проста тому, що МДН-транзистор є єдиним типовим елементом, а між іншими елементами використовуються безпосередні зв'язки.

Проектуючи топологію, необхідно мінімізувати площу кристала, але при цьому необхідно одержати нормальні характеристики мікросхем. Статичні і динамічні характеристики МДН-мікросхем значно залежать від паразитних ємностей і опорів, зумовлених топологією.

Сумарна паразитна ємність елемента АБО-НЕ:

$$C_{\Pi} = M_0 C_{3K1} + 2M_0 C_{3C1} + C_{3B0} + C_M + C_H,$$

де C_{3K1} - ємність між затвором і каналом керуючого транзистора VT1;

C_{3C1} - ємність між затвором і стоком керуючого транзистора

($C_{3C1} \approx C_{3B1}$);

C_{3B0} - ємність між затвором і витоком навантажувального транзистора VT0;

C_M - паразитна ємність металевих з'єднань та ізолюючого *p-n* - переходу об'єднаної області стоків транзисторів VT1 і витоку транзистора VT0;

C_H - ємність навантаження.

Ємність C_{3K1} обчислюється за формулою:

$$C_{3K1} = C_0 L_{KAN1} B_{KAN1}.$$

Ємності $C_{3C1} \approx C_{3B1}$ виникають через існування перекриття затвором областей стоку і витоку:

$$C_{3C1} \approx C_{3B1} = C_0 l_{nep} B_{KAN1},$$

де l_{nep} - довжина перекриття.

Ємність $C_{зб0}$ визначають аналогічно:

$$C_{зб0} = C_0 l_{пер} B_{КАН0}.$$

Ємності C_M і C_H задаються у вихідних даних.

У МДН-мікросхемах використовуються прямокутні конфігурації транзисторів, що відрізняються лише різними відношеннями ширини до довжини каналу в залежності від необхідного значення крутості стоко-затворної характеристики транзистора, що визначає його підсилювальні властивості.

Крутість транзистора в області насичення:

$$S = \frac{\partial I_C}{\partial U_{зб}} \Big|_{U_{св} = U_{зб} - U_{пор}} = \mu_n C_0 \frac{B_{КАН}}{L_{КАН}} U_{с НАС} = b \cdot U_{с НАС}, \quad (7.2)$$

де b - питома крутість.

З формули (7.2) очевидно, що для збільшення крутості необхідно зменшувати товщину окислу під затвором, зменшувати довжину каналу $L_{КАН}$ і збільшувати ширину каналу $B_{КАН}$. Прилади з більшою крутістю повинні мати широкий канал.

Швидкодія МОН-транзисторів (табл. 7.2) визначається часом прольоту носіїв у каналі і тривалістю процесів заряду і розряду паразитної ємності. Щоб зменшити час прольоту і ємність «затвор-канал», необхідно зменшити довжину каналу. У той же час мала ємність $C_{зк} = C_0 L_{КАН} B_{КАН}$ повинна бути забезпечена при достатньо великій питомій ємності $C_0 = \varepsilon_0 \varepsilon_d / d$ структури, яка визначає статичні параметри МОН-транзистора.

У схемі логічного елемента АБО-НЕ (рис. 7.4) транзистор VT_0 постійно відкритий, тому його параметри, що визначають швидкодію, не є визначальними. Отже, довжину каналу навантажувального транзистора VT_0 можна збільшувати відповідно вимогам до статичних параметрів.

Конструювання логічного елемента АБО-НЕ спирається на методику розрахунку, в основу якої лежить вибір співвідношень розмірів каналів керуючого VT_1 і навантажувального VT_0 транзисторів, виходячи з вимог завадостійкості. Потім обрана топологія коректується, якщо не виконуються умови заданої швидкодії.

Порогові напруги навантажувального і керуючих транзисторів приймаються рівними.

Розміри МОН-транзисторів, виходячи з вимог завадостійкості, вибираються так, щоб забезпечити достатньо великий логічний перепад напруги

$$\Delta U_{\text{л}} = U^1 - U^0.$$

Опір МОН-транзистора, що знаходиться у відкритому стані, обернено пропорційний крутості b , тобто прямо пропорційний відношенню $L_{\text{кан}} / B_{\text{кан}}$.

Коли один або обидва керуючі транзистори відкриті, тобто коли в схемі тече струм, необхідно, щоб опір навантажувального транзистора $VT0$ був набагато більше опорів відкритих керуючих транзисторів ($\approx$ у 10...20 разів). У противному випадку напруга логічного нуля на виході буде недостатньо низькою, а $\Delta U_{\text{л}}$ занадто малим.

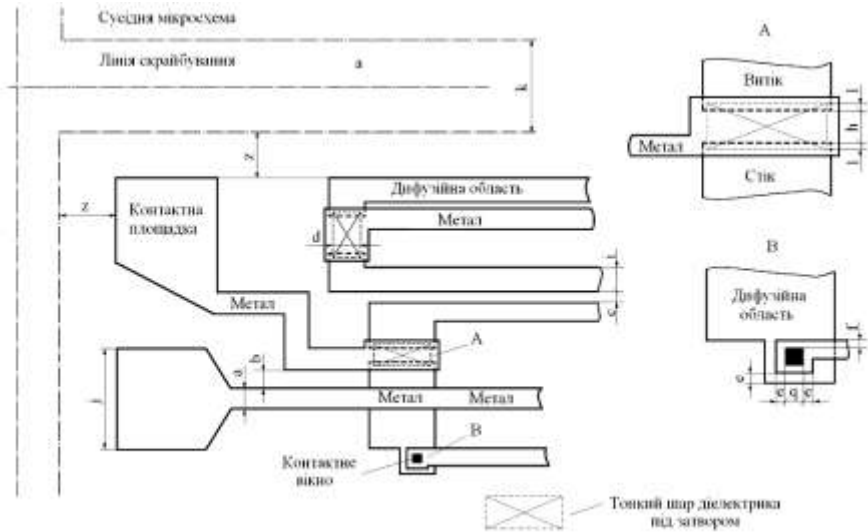
Отже, відношення $L_{\text{кан } 0} / B_{\text{кан } 0}$ навантажувального транзистора $VT0$ має бути більше співвідношення $L_{\text{кан } 1} / B_{\text{кан } 1}$ керуючого транзистора (тобто питому крутість b_0 транзистора $VT0$ необхідно зменшувати, а питому крутість b_1 , транзистора $VT1$ необхідно збільшувати).

Отже, для підвищення завадостійкості необхідно збільшувати $L_{\text{кан } 0}$, зменшувати $B_{\text{кан } 0}$, зменшувати $L_{\text{кан } 1}$, збільшувати $B_{\text{кан } 1}$, тобто навантажувальні транзистори необхідно виконувати з вузькими і довгими каналами, а керуючі - з широкими і короткими.

Для реалізації МДН-мікросхем використовуються різноманітні методи планарної технології. Технологічні можливості конкретного процесу виготовлення МДН-мікросхем накладають ряд обмежень на проектування її топології. На рис. 7.5 показана частина топологічного креслення МДН-мікросхеми, а також приведені позначення мінімальних відстаней між елементами і їхні орієнтовані значення.

Щільність розміщення елементів ВІС на кристалі підвищують за рахунок застосування функціональної інтеграції, що дозволяє сполучати напівпровідникові області одного типу провідності на кристалі.

Топологія логічного елемента АБО-НЕ, електрична схема якого приведена на рис. 7.4, показана на рис. 7.6. Сполученими областями є області витоків керуючих транзисторів $VT1$ і область витoku навантажувального транзистора $VT0$.



- a - мінімальна ширина металізації (до 5 мкм);
- b - мінімальний зазор між лініями металізації (до 5 мкм);
- i - мінімальна ширина дифузійної області (до 5 мкм);
- c - мінімальний зазор між дифузійними областями (до 5 мкм);
- d - мінімальна ширина затвора, рівна ширині тонкого оксиду (до 5 мкм);
- h - мінімальна довжина каналу (до 5 мкм);
- l - мінімальне перекриття затвором дифузійних областей (0-2 мкм);
- q - мінімальний розмір вікна в оксиді під контакт (до 5 мкм);
- e - мінімальна відстань від контактного вікна до краю дифузійної області (до 3 мкм);
- f - мінімальний вихід металевого шару за межі контактного вікна з усіх боків (до 2 мкм);
- j - мінімальна довжина і ширина контактної площадки (до 60 мкм);
- z - мінімальна відстань від краю лінії скрайбування до контактної площадки (до 50 мкм);
- k - розмір ширини лінії скрайбування (до 75 мкм).

Рисунок 7.5 - Фрагмент спільного креслення топології мікросхеми

Конструкція МОН-транзистора з алюмінієвим затвором є найбільш простою. За межами МОН-транзистора кремнієвий підшарок вкритий товстим окислом (до 1,5 мкм) для зменшення паразитної ємності сигнальних шин, зниження браку через проколи в окислі, підвищення граничної напруги паразитних МОН-транзисторів, що утворю-

ються на ділянках проходження алюмінієвих шин над шинами живлення і заземлення, якщо такі є в структурі.

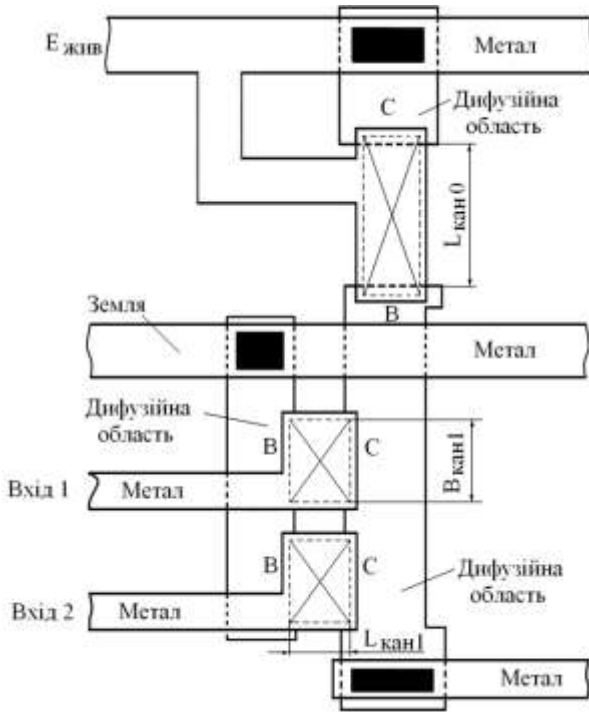
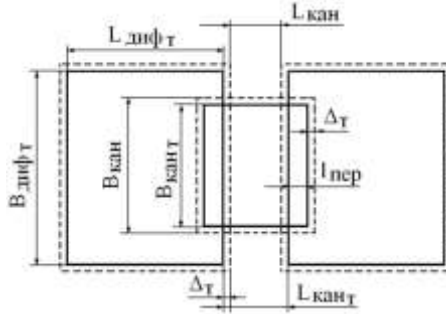


Рисунок 7.6 - Топологія логічного елемента АБО-НЕ

Топологія МОН-транзистора з урахуванням зміни геометричних розмірів елемента в процесі виготовлення елемента показана на рис. 7.7.

Алгоритм автоматизованого конструювання логічного елемента АБО-НЕ на МОН-транзисторах приведений на рис. 7.8. Крім згадуваних раніше, в алгоритмі використане позначення $\Delta_{\min}$ мінімальний розмір вікна на фотошаблоні.

У блоках 3-7 за допомогою зміни концентрації вихідної домішки в підшарку і зміни внаслідок цього порогової напруги розраховуються логічні рівні нуля й одиниці на виході логічного елемента, що задовольняють умовам завадостійкості.



$L_{КАН\ T}, B_{КАН\ T}$ - топологічні (що закладаються у фотошаблон) розміри каналу;

Δ_T - відхід розміру при виконанні технологічних операцій;

$L_{ДИФ\ T}, B_{ДИФ\ T}$ - топологічні розміри областей стоку і витоку;

$l_{ПЕР}$ - довжина перекриття затвора з витоком і стоком;

$L_{КАН} = L_{КАН\ T} + 2\Delta_T$ - істинна довжина каналу;

$B_{КАН} = B_{КАН\ T} + 2\Delta_T$ - істинна ширина каналу

Рисунок 7.7 - Топологія інтегрального МОН-транзистора

Формули для розрахунку параметрів логічного елемента беруться з табл. 7.2. Порогова напруга розраховується за формулою (7.1).

Коефіцієнт K визначається з формули:

$$K = \frac{b_1}{b_0} = \frac{5(E_{жив} - U_{пор})^2}{M_0 U_{пор} (E_{жив} - 2U_{пор})}. \quad (7.3)$$

У блоці 10 алгоритму розміри каналів $B_{КАН\ 0}$ і $L_{КАН\ 1}$ приймаються рівними мінімально можливим (рис. 7.8). Розміри каналів $B_{КАН\ 1}$ і $L_{КАН\ 0}$ приймаються однаковими і визначаються, виходячи з розрахованого за формулою (7.3) значення коефіцієнта K . Тому що

$$K = \frac{b_1}{b_0} = \left(\mu_n C_0 \frac{B_{КАН1}}{L_{КАН1}} \right) / \left(\mu_n C_0 \frac{B_{КАН0}}{L_{КАН0}} \right) = \frac{B_{КАН1} L_{КАН0}}{(\Delta_{min} + 2\Delta_T) \cdot (\Delta_{min} - 2\Delta_T)} = \frac{B_{КАН1} L_{КАН0}}{\Delta_{min}^2 - 4\Delta_T^2},$$

розміри каналів визначаються зі співвідношення

$$B_{КАН1} = L_{КАН0} = Z = \sqrt{K(\Delta_{min}^2 - 4\Delta_T^2)}.$$

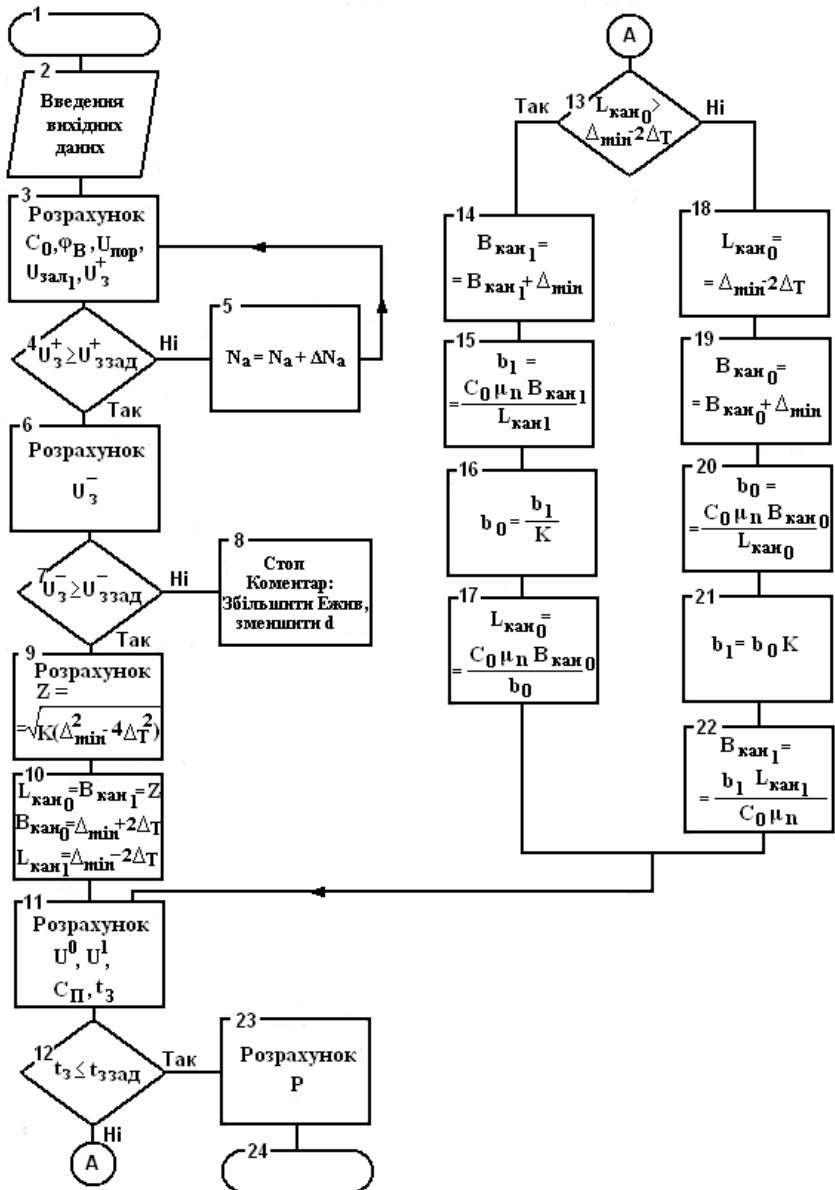


Рисунок 7.8 - Алгоритм автоматизованого конструювання логічного елемента АБО-НЕ

Якщо розрахована середня затримка перемикавання при обраних розмірах каналів стане більше заданої (блок 12), то проводиться корекція розмірів каналів (блоки 13-22) доти, поки зазначена затримка не стане менше (або рівною) заданої затримки.

7.2 Завдання

Відповідно до наведеного алгоритму (рис. 7.8) складіть програму автоматизованого конструювання логічного елемента АБО-НЕ.

Проведіть топологічний розрахунок логічного елемента АБО-НЕ на МОН-транзисторах.

Вихідні параметри, значення яких наведено у додатку Г, мають наступні позначення: напруга живлення логічного елемента $E_{жив}$; додатна завадостійкість $U_{зЗД}^+$; від'ємна завадостійкість $U_{зЗД}^-$; середня затримка перемикавання $t_{зЗД}$; кількість входів логічного елемента M_0 ; густина сумарного позитивного поверхневого заряду Q_{SS} ; мінімальна початкова концентрація акцепторної домішки в підшарку N_{Amin} ; крок збільшення концентрації акцепторної домішки ΔN_A ; мінімальний розмір вікна на фотошаблоні Δ_{min} ; відхід розміру при виконанні технологічних операцій Δ_T ; паразитна ємність металевих з'єднань та ізолюючого $p-n$ переходу об'єднаної області стоків транзисторів VT1 і витоку транзистора VT0 C_M ; ємність навантаження C_H ; товщина діелектрика під затвором d ; довжина перекриття затвором областей стоку і витоку $l_{ПЕР}$;

Необхідні для розрахунку електрофізичні сталі: діелектрична проникність вакууму $\epsilon_0 = 8,86 \cdot 10^{-14}$ Ф/см; відносна діелектрична проникність SiO_2 $\epsilon_d = 3,5$; приповерхнева рухливість електронів $\mu_n = 550 \text{ см}^2/(\text{В} \cdot \text{с})$. При розрахунках прийняти $U_{зЗД} = 0,1U_{ПОР}$.

7.3 Зміст звіту

Звіт розрахункового завдання має містити: енергетичні діаграми структури МДН; модель МОН - транзистора; алгоритм розрахунку порогової напруги МОН - транзистора; схему електричну логічного

елемента АБО-НЕ; формули розрахунку електричних і часових параметрів ЛЕ АБО-НЕ; топологію логічного елемента АБО-НЕ.

7.4 Контрольні запитання

- 1 Які позитивні якості мають ІМС на МОН-структурах?
- 2 Який принцип дії МОН-транзистора?
- 3 Чому виникає поверхневий заряд у напівпровіднику при відсутності потенціалів на витоках МДН-транзистора? Який його знак?
- 4 Внаслідок чого виникає контактна різниця потенціалів між металом і напівпровідником?
- 5 Як утворюється інверсна область у напівпровіднику в МОН-структурі?
- 6 Який вид мають статичні характеристики МДН-транзистора з індукованим каналом?
- 7 Які елементи функціональної інтеграції є в елементі АБО-НЕ?
- 8 Які паразитні ємності є в елементі АБО-НЕ?
- 9 Що таке питома крутість транзистора?
- 10 Навіщо канали навантажувальних транзисторів виготовляються довгими і вузькими?
- 11 Який принцип роботи має логічний елемент АБО-НЕ?
- 12 Як підвищують щільність розміщення елементів ВІС на кристалі з МОН-структурами?
- 13 На що впливає величина довжини перекриття затвором областей стоку і витоку?
- 14 Які електричні та часові параметри логічного елемента розраховуються в алгоритмі?
- 15 Від чого залежить значення контактної різниці потенціалів $q\psi_{MS}$ в МОН-структурі?
- 16 Якими процесами в МОН-транзисторі визначається його швидкодія?

8 ПЕРЕЛІК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

- 1 **Грушвицкий, Р.И.** Проектирование систем на микросхемах программируемой логики [Текст] / Р.И. Грушвицкий, А.Х. Мурсаев, Е.П. Угрюмов – СПб.: БХВ-Петербург, 2002. – 608 с.
- 2 **Бибило, П.Н.** Синтез логических схем с использованием языка VHDL [Текст] / П.Н. Бибило – М.: СОЛОН-Р, 2002. – 384 с.
- 3 **Комолов, Д.А.** Системы автоматизированного проектирования фирмы Altera MAX+plus II и Quartus II. Краткое описание и самоучитель [Текст] / Д.А. Комолов, З.А. Мяльк, А.А. Зобенко, А.С. Филиппов – М.: ИП РадиоСофт, 2002. – 352 с.
- 4 **Уилкинсон, Б.** Основы проектирования цифровых схем [Текст] / Б. Уилкинсон – М.: Издательский дом «Вильямс», 2004. – 320 с.
- 5 **Пономарев, М.Ф.** Конструирование и расчет микросхем и микропроцессоров [Текст] / М.Ф. Пономарев, Б.Г. Коноплев - М.: Радио и связь, 1986.- 176 с.
- 6 **Пономарев, М.Ф.** Конструкции и расчет микросхем и микроэлементов ЭВА [Текст] / М.Ф. Пономарев - М.: Радио и связь, 1982. - 288 с.

Додаток А

VHDL – код, який реалізує алгоритм роботи системи S

```

entity vlsi_1 is
port (a2,a1,b2,b1,x: in BIT;
      d4,d3,d2,d1: out BIT);
end vlsi_1;
architecture structure of vlsi_1 is
component adder_2
port (a1,b1,a2,b2: in BIT;
      c2,s2,s1: out BIT);
end component;
component mult_2
port (s1,s0,r1,r0: in BIT;
      t3,t2,t1,t0: out BIT);
end component;
component dd
port (x1,x2,x3,x4,x5,x6: in BIT;
      y1,y2,y3: out BIT);
end component;
component yy
port (a2,a1,b2,b1,x: in BIT;
      f6,f5,f4,f3,f2,f1: out BIT);
end component;
signal f1,f2,f3,f4,f5,f6,t4,t3,t2,t1, c2, s2, s1: BIT; --декларація внутрішніх сигналів
begin

    circ1: yy
    port map(a2,a1,b2,b1,x,f6,f5,f4,f3,f2,f1);
    circ2: mult_2
    port map(b2,b1,f2,f1,d4,t3,t2,t1);
    circ3: adder_2
    port map(f4,f3,f6,f5,c2,s2,s1);
    circ4: dd
    port map(s1,t1,s2,t2,c2,t3,d1,d2,d3);

end structure;
entity adder_2 is
port (a1,b1,a2,b2: in BIT;
      c2,s2,s1: out BIT);
end adder_2;
architecture structure of adder_2 is
component
add1
port (b1,b2: in BIT;
      c1,s1: out BIT);
end component;
component add2
port (c1,a1,a2: in BIT;
      c2,s2: out BIT);
end component;
signal c1: BIT;
begin

    circ1: add1
    port map(b1,b2,c1,s1);
    circ2: add2
    port map(c1,a1,a2,c2,s2);

end structure;
entity mult_2 is
port (s1,s0,r1,r0: in BIT;

```

--VHDL – код для опису системи S

--декларація компонента

--декларація компонента

--декларація компонента

--декларація компонента

--опис об'єкта adder_2

-- опис об'єкта mult_2

```

    t3,t2,t1,t0: out BIT);
end mult_2;
architecture structure of mult_2 is
component
add1 port (b1,b2: in BIT;
          c1,s1: out BIT);
end component;
signal p1,p2,p3,p4: BIT;
begin
    t0 <= r0 and s0;
    p2 <= r0 and s1;
    p1 <= r1 and s0;
    p4 <= r1 and s1;
    circ1: add1 port map (p1,p2,p3,t1);
    circ2: add1 port map (p3,p4,t3,t2);
end structure;
entity dd is
port (x1,x2,x3,x4,x5,x6: in BIT;
      y1,y2,y3: out BIT);
end dd;
architecture struct_2 of dd is
begin
    y1 <= x1 or x2;
    y2 <= x3 or x4;
    y3 <= x5 or x6;
end struct_2;
entity add1 is
port (b1, b2: in BIT;
      c1, s1: out BIT);
end add1;
architecture struct_3 of add1 is
begin
    s1 <= ((b1 and (not b2)) or ((not b1) and b2));
    c1 <= b1 and b2;
end struct_3;
entity yy is
port (a2,a1,b2,b1,x: in BIT;
      f6,f5,f4,f3,f2,f1: out BIT);
end yy;
architecture struct_4 of yy is
begin
    f1 <= x and a1;
    f2 <= x and a2;
    f3 <= not x and a1;
    f4 <= not x and a2;
    f5 <= not x and b1;
    f6 <= not x and b2;
end struct_4;
entity add2 is
port (c1, a1, a2: in BIT;
      c2, s2: out BIT);
end add2;
architecture struct_6 of add2 is
begin
    s2<=((not c1) and (not a1) and a2)or((not c1) and a1 and (not a2)) or (c1 and (not a1) and (not a2)) or (c1 and a1 and
a2) ;
    c2<=((not c1) and a1 and a2) or ( c1 and (not a1) and a2) or (c1 and a1 and (not a2)) or (c1 and a1 and a2) ;
end struct_6;

```

-- элемент el_1;
-- элемент el_3;
-- элемент el_2;
-- элемент el_4;

-- опис объекта dd

-- опис объекта add1

-- опис объекта yy

-- опис объекта add2

Додаток Б

Автомат Мілі з п'ятьма станами

```

library ieee;      -- Автомат Мілі з 5 станами
use ieee.std_logic_1164.all;
entity mealy is
port (clock, reset: in std_logic;
data_out: out std_logic;
data_in: in std_logic_vector (1 downto 0));
end mealy;
architecture behave of mealy is
type state_values is (st0, st1, st2, st3, st4);
signal pres_state, next_state: state_values;
begin
statereg: process (clock, reset)  --FSM perictrp
begin
if (reset = '0') then
pres_state <= st0;
elsif (clock'event and clock = '1') then
pres_state <= next_state;
end if;
end process statereg;
fsm: process (pres_state, data_in)
begin
case pres_state is
when st0 =>
case data_in is
when "00" => next_state <= st0;
when "01" => next_state <= st4;
when "10" => next_state <= st1;
when "11" => next_state <= st2;
when others => next_state <= st0;
end case;
when st1 =>
case data_in is
when "00" => next_state <= st0;
when "10" => next_state <= st2;
when others => next_state <= st1;
end case;
when st2 =>
case data_in is
when "00" => next_state <= st1;
when "01" => next_state <= st1;
when "10" => next_state <= st3;
when "11" => next_state <= st3;
when others => next_state <= st0;
end case;
when st3 =>
case data_in is
when "01" => next_state <= st4;
when "11" => next_state <= st4;
when others => next_state <= st3;
end case;
when st4 =>

```

```

case data_in is
when "11" => next_state <= st4;
when others => next_state <= st0;
end case;
when others => next_state <= st0;
end case;
end process fsm;
outputs: process (pres_state, data_in)
begin
case pres_state is
when st0 =>
case data_in is
when "00" => data_out <= '0';
when others => data_out <= '1';
end case;
when st1 => data_out <= '0';
when st2 =>
case data_in is
when "00" => data_out <= '0';
when "01" => data_out <= '0';
when others => data_out <= '1';
end case;
when st3 => data_out <= '1';
when st4 =>
case data_in is
when "10" => data_out <= '1';
when "11" => data_out <= '1';
when others => data_out <= '0';
end case;
when others => data_out <= '0';
end case;
end process outputs;
end behave;

```

Додаток В

Варіанти завдань до практичного заняття №6

Таблиця В.1 - Варіанти завдань до практичного заняття №6

[illegible]

Продовження табл. В.1

[illegible]

Продовження табл. В.1

[illegible]

Додаток Г

Варіанти завдань до практичного заняття №7

Таблиця Г.1 - Варіанти завдань до практичного заняття №7

[illegible]

[illegible]

Продовження табл. Г.1

[illegible]