МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ТЕХНОЛОГІЇ КОМП'ЮТЕРНОГО ПРОЕКТУВАННЯ

Комп'ютерний практикум

Навчальний посібник

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для здобувачів ступеня бакалавра
за освітньою програмою «Інтелектуальні сервіс-орієнтовані розподілені обчислювання»
спеціальності 122 Комп'ютерні науки

Електронне мережне навчальне видання

Київ
КПІ ім. І. Сікорського
2022

Укладачі: Артюхов В.Г., к.т.н, доцент
Бритов О.А.
Гіоргізова-Гай В.Ш., к.т.н.
Кирюша Б.А., к.т.н.
Стіканов В.Ю., к.т.н., с.н.с.

Рецензент Шумков Ю.С., к.т.н., доцент, кафедра інформаційно-вимірювальних технологій приладобудівного факультету КПІ ім. Ігоря Сікорського

Відповідальний редактор Мельник І.В., д.т.н., професор

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського
(протокол № 1 від 02.09.2022р.)
за поданням Вченої ради навчально-наукового інституту прикладного системного аналізу
(протокол № 7 від 01.09.2022 р.)

Посібник містить методичні вказівки до виконання комп'ютерних практикумів з дисципліни «Технології комп'ютерного проектування» у предметній області проектування цифрових електронних пристроїв з використанням мови VHDL.

Цикл лабораторних робіт включає побудову різних типів моделей для стандартних елементів цифрових пристроїв засобами мови VHDL та проведенню аналізу їх роботи в середовищі сучасних САПР. Додатки містять довідковий матеріал з курсу «Комп'ютерна схемотехніка», достатній для виконання індивідуальних варіантів завдань.

Посібник призначений для здобувачів ступеня бакалавра за спеціальністю 122 «Комп'ютерні науки», а також для всіх, хто використовує мову VHDL у своїй практичній діяльності.

Реєстр. № НП 22/23-043. Обсяг 3,3 авт. арк.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
проспект Перемоги, 37, м. Київ, 03056
https://kpi.ua
Свідоцтво про внесення до Державного реєстру видавців, виготовлювачів і розповсюджувачів видавничої продукції ДК № 5354 від 25.05.2017 р.

ЗМІСТ

ПЕРЕДМОВА

Розробка складних комп'ютерних систем і їх вузлів сьогодні практично неможлива без засобів автоматизації проектних процедур і систем комп'ютерного проектування. Сучасний процес проектування цифрових пристроїв (ЦП) потребує точного і вичерпного опису проектних рішень, постійної перевірки їх відповідності вимогам технічного завдання на всіх етапах проектування. Впродовж маршруту проектування процедури ручного та автоматичного синтезу постійно чергуються з процедурами аналізу та верифікації, які проводяться шляхом моделювання.

Стандартним засобом вирішення цих задач є мови опису апаратури (Hardware Description Languages - HDL), такі як Verilog та VHDL. Ці мови застосовуються для формального опису проектів, підтримуються як вхідні мови для переважної більшості систем автоматизованого проектування (САПР), використовуються як для підсистем моделювання, так і автоматичного синтезу ЦП. Таким чином, вони є базовими під час розробки апаратури сучасних обчислювальних систем на програмованих логічних інтегральних схемах ПЛІС і у вигляді спеціалізованих інтегральних схем – ASIC (application-specific integrated circuit, інтегральних схем для конкретного застосування.

Метою комп'ютерного практикуму є:

1. Краще засвоєння студентами теоретичного матеріалу дисципліни «Технології комп'ютерного проектування» з розділів, присвячених:
   - класифікації моделей і методів моделювання;
   - аналізу переваг і обмежень моделей в залежності від задач, які ставляться на різних етапах проектування ЦП;
   - побудові моделей типових вузлів ЦП засобам мови VHDL.
2. Придбання навиків застосуванням систем автоматизованого проектування при розробці цифрових пристроїв, навиків створення моделей типових елементів та вузлів сучасної цифрової апаратури засобами мови VHDL і проведення аналізу їх роботи в середовищі сучасних САПР.

Вважається за необхідне наявність у студентів знань з дисциплін: «Дискретна математика», «Комп'ютерна схемотехніка», «Основи алгоритмізації та програмування».

ВСТУП

Сьогодні VHDL (Very high speed integrated circuits Hardware Description Language), що відповідає стандарту ANSI/IEEE Std 1067-1993, є найбільш поширеною мовою опису апаратури в країнах Європи і США.

Відповідно до загальної класифікації мова VHDL підтримує два типи моделей – поведінкові і структурні. Поведінкові моделі описують функціонування об'єкту у вигляді явних або неявних залежностей вихідних параметрів від вхідних і внутрішніх, тоді як структурні моделі описують систему у вигляді сукупності компонентів (блоків і елементів) і зв'язків між ними. Поведінкові моделі в свою чергу можна розділити на аналітичні – логічно-арифметичні рівняння, табличні – таблиці істинності і алгоритмічні – алгоритм перетворення вхідних даних у вихідні.

Загальна модель пристрою представляє собою структуру, яка на нижчому ієрархічному рівні складається із сукупності поведінкових моделей елементів пристрою, з'єднаних між собою лініями зв'язку – сигналами. Кожна поведінкова модель представляє собою незалежний процес обробки даних. Всі процеси протікають паралельно в модельному часі і описується окремими паралельними операторами. С точки зору обчислювальної моделі VHDL загальна модель пристрою є множиною паралельних операторів, з'єднаних між собою сигналами. Порядок активації паралельних операторів визначається моментами часу, в які змінюються значення сигналів на входах того чи іншого процесу. Загальна структурна модель може бути багаторівневою, і на вищих рівнях ієрархії складатись зі структурних моделей нижчих рівнів, які з'єднуються між собою сигналами.

Відповідно до прийнятої у VHDL термінології підтримуються три різні стилі опису апаратних архітектур: потоковий, поведінковий і структурний.

При потоковому описі (data-flow description) архітектура пристрою представляється у вигляді паралельного потоку обробки вхідних сигналів, і

зазвичай складається з сукупності паралельних операторів привласнення значень сигналам (<=) та виклику процедур, якими описуються різні поведінкові моделі (аналітичні, табличні та алгоритмічні).

При поведінковому описі (behavioral description) архітектура пристрою представляється у вигляді одного або сукупності незалежних, але синхронізованих у часі процесів, які описують алгоритм функціонування пристрою. Такий опис складається з одного або декількох паралельних операторів Process. Оператор Process може містить алгоритмічну модель пристрою або одного з його блоків, яка за допомогою сукупності послідовних операторів структурного програмування і тимчасових змінних описує алгоритм обчислення значень вихідних сигналів в залежності від вхідних сигналів.

При структурному описі (structural description) архітектура пристрою представляється у вигляді ієрархії зв'язаних сигналами компонентів, описаних своїми інтерфейсами. Моделі окремих вузлів та різних типів елементів попередньо записуються у бібліотеку проекту і потім вибираються як компоненти його структури. Для опису моделей може бути обрана будь-яка з перерахованих форм. Структурний опис складається з набору паралельних операторів реалізації екземплярів компонентів. Порти окремих екземплярів з'єднуються між собою через сигнали за допомогою карти портів (port map). Кількість рівнів вкладеності структурної моделі може бути довільна.

У VHDL припускається змішаний стиль побудови моделі проекту, в якому всі перераховані стилі можуть спільно використатися при описі архітектури проекту. Гарним стилем програмування вважається, коли в ієрархічному проекті об’єкти верхніх рівнів описуються структурним стилем.

Мовні засоби VHDL дозволяють створювати моделі різного ступеня деталізації для різних задач проектування: синтезу, аналізу та верифікації. Моделі, призначені для аналізу роботи проекту і пошуку можливих помилок, можуть бути більш детальними і використовувати всі конструкції мови. Підсистеми автоматичного синтезу підтримують обмежену множину

конструкції VHDL. Тому, перед передачею моделі проекту певному синтезатору з неї видаляються елементи коду, які були призначені для тестування проекту.

Цикл лабораторних робіт включає побудову моделей різних типів для стандартних елементів цифрових пристроїв засобами мови VHDL та проведенню аналізу їх роботи в середовищі сучасних САПР.

ЛАБОРАТОРНА РОБОТА № 1
ПОБУДОВА АНАЛІТИЧНИХ МОДЕЛЕЙ КОМБІНАЦІЙНИХ СХЕМ НА МОВІ VHDL

1.1. Мета роботи

- ознайомлення з інтерфейсом і принципами роботи системи проектування цифрових схем;
- засвоєння основних понять мови VHDL, загальної структури опису цифрового пристрою, методики формування вхідних даних;
- побудова простіших моделей, описаних за допомогою паралельних операторів безумовного призначення значень сигналам, і перевірка правильності їх роботи.

1.2. Рекомендації

Для виконання роботи необхідно повторити і ознайомитись з наступним теоретичним матеріалом:

- булеві функції, таблиці істинності, логічні вентилі («Додаток А»);
- засоби опису комбінаційних схем у вигляді систем булевих рівнянь у мові VHDL (скалярні сигнали типу bit, паралельний оператор безумовного призначення значення сигналу) [1–4];
- принципами роботи в обраному пакеті САПР цифрових пристроїв.

1.3. Завдання

1.3.1) Ознайомтесь з теоретичними відомостями.

1.3.2) Виписати функції Ф1 і Ф2 за варіантом завдання. Номер варіанту відповідає номеру студента у списку групи. У таблиці 1.1 вказано номери функцій Ф1 і Ф2 від трьох змінних {a, b, c} у переліку логічних функцій:

Функція Ф1:

1. Ф1(a, b, c) = (a v b)’ & c;
2. Ф1(a, b, c) = a v (b & c);
3. Ф1(a, b, c) =-a’ v (b & c);
4. Ф1(a, b, c) =(a v b v c)’;
5. Ф1(a, b, c) = (a’ v b’)’ & c;
6. Ф1(a, b, c) = a & b & ‘c;
7. Ф1(a, b, c) = (a & b)’ v c.
8. Ф1(a, b, c) = (a & b)’ v (c & b)’;
9. Ф1(a, b, c) = (a v b) & (c v b)’;
10. Ф1(a, b, c) = (a & b)’ & c’;
11. Ф1(a, b, c) = a’ v b v c;
12. Ф1(a, b, c) = a v (c’ & b);

Функція Ф2:

1. Ф2(a, b, c) = (a & b) & c’;
2. Ф2(a, b, c) = a v (b v c)’;
3. Ф2(a, b, c) = a’ & b’ & c’;
4. Ф2(a, b, c) = (b & c) v (c’ & a);
5. Ф2(a, b, c) = (a & b)’ & c;
6. Ф2(a, b, c) = (a & b & c) v a v b;
7. Ф2(a, b, c) = (a & b)’ & c v a.
8. Ф2(a, b, c) = (c & b)’ & a’.
9. Ф2(a, b, c) = a’ v b’ v c’;
10. Ф2(a, b, c) = a & (c & b)’;
11. Ф2(a, b, c) = a’ & c & b’;
12. Ф2(a, b, c) =(a’ & b’)v c;

Таблиця 1.1. Варіанти завдань до лабораторної роботи №1

Варіант	Номери логічних функцій		Варіант	Номери логічних функцій	
	Ф1	Ф2		Ф1	Ф2
1	1	1	31	4	12
2	1	2	32	4	5
3	1	3	33	4	6
4	1	4	34	4	7
5	1	5	35	4	8
6	1	6	36	4	9
7	1	7	37	4	10
8	1	8	38	4	11
9	1	9	39	10	11
10	1	10	40	5	6
11	1	11	41	5	7
12	2	12	42	5	8
13	2	3	43	5	9
14	2	4	44	5	10

Варіант	Номери логічних функцій		Варіант	Номери логічних функцій	
15	2	5	45	5	11
16	2	6	46	6	12
17	2	7	47	6	7
18	2	8	48	6	8
19	2	9	49	6	9
20	2	10	50	6	10
21	2	11	51	6	11
22	3	12	52	9	11
23	3	4	53	7	8
24	3	5	54	7	9
25	3	6	55	7	10
26	3	7	56	7	11
27	3	8	57	8	9
28	3	9	58	8	10
29	3	10	49	8	11
30	3	11	60	9	10

1.3.3) Намалювати схему у вигляді принципової вентильної схеми згідно рис.1.1. Базис реалізації довільний. Заповнити вручну таблицю істинності схеми для подальшого порівняння з результатами комп’ютерного моделювання.

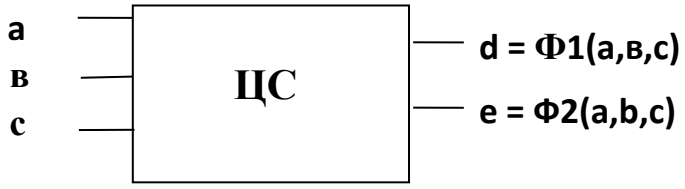


Рис. 1.1. Загальний вигляд досліджуваної ЦС

1.3.4) Описати розроблену схему на мові VHDL у вигляді булевих рівнянь з введенням проміжних сигналів за допомогою паралельного оператора безумовного призначення значень сигналам та скалярного типу bit.

1.3.5) Для проведення моделювання цифрової схеми на її входи необхідно подати $N=2^3=8$ різних логічних наборів вхідних сигналів. Нехай тривалість тактового імпульсу буде 50 у.о., тоді час моделювання має бути більше ніж 50 * 8 = 400 у.о. Для завдання двійкового коду {a, b, c} = { (0,0,0),..,(1,1,1) } (розряд a - молодший) на входи ЦС зручно подати послідовності вхідних сигналів, що зображено на рис.1.2.

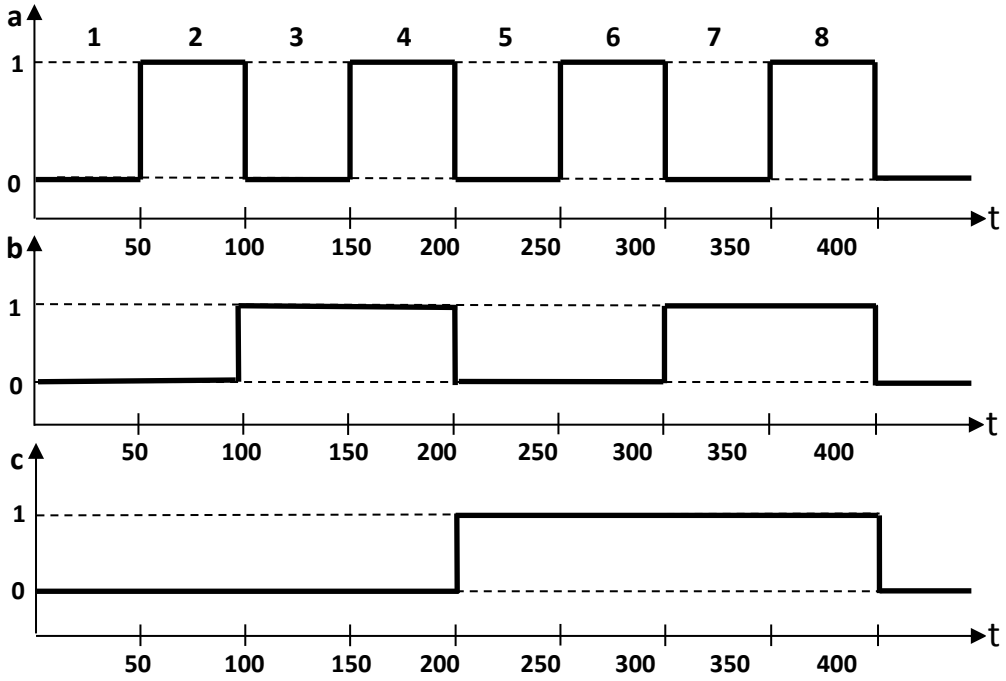


Рис. 1.2. Часові діаграми для вхідних логічних сигналів

1.3.6) Виконайте моделювання ЦС з нульовими затримками сигналів. Дослідіть результати моделювання у формі часових діаграм і у формі таблиць істинності.

1.4. Зміст звіту

1.4.1) Мета роботи.

1.4.2) Завдання за варіантом.

1.4.3) Принципова (вентильна) схема і таблиця істинності.

1.4.4) Текст програми і результати моделювання у табличній формі і у формі часових діаграм.

1.4.5) Висновки по роботі.

1.5. Контрольні запитання

1.5.1) Розповісти принцип заповнення таблиць істинності (згадати булеву алгебру).

1.5.2) Показати роботу з інтерфейсом системи проектування цифрових схем: створення проекту, опис схеми, введення і виведення результатів, організація моделювання на своєму комп’ютері.

1.5.3) Розповісти про загальну структуру опису цифрового пристрою і методику формування вхідних даних у мові VHDL на прикладі своєї схеми.

1.5.4) Що таке паралельні і послідовні оператори у мови VHDL (на прикладі своєї схеми)?

1.5.5) Як визначається порядок виконання паралельних операторів у мови VHDL? В які моменти часу це буде відбуватися на прикладі своєї схеми.

1.5.6) Показати оператори, операції, сигнали на прикладі своєї схеми.

ЛАБОРАТОРНА РОБОТА № 2
ПОБУДОВА ТАБЛИЧНИХ МОДЕЛЕЙ КОМБІНАЦІЙНИХ СХЕМ НА МОВІ VHDL

2.1. Мета роботи

- побудова табличних моделей комбінаційних схем і перевірка правильності їх роботи;
- придбання досвіду використання паралельних операторів умовного та вибіркового призначення значень сигналам і векторної форми представлення сигналів у мові VHDL.

2.2. Рекомендації

Для виконання роботи необхідно повторити і ознайомитись з наступним теоретичним матеріалом:

- принципи роботи двійкових мультиплексорів, демультиплексорів, шифраторів і дешифраторів, їх таблиці істинності, умовні графічні позначення («Додаток Б»);
- засоби опису комбінаційних схем у вигляді таблиць істинності у мові VHDL (векторні сигнали типу bit_vector, паралельні оператори умовного та вибіркового призначення значення сигналу) [1–4];

2.3. Завдання

2.3.1) Ознайомтесь з теоретичними відомостями.

2.3.2) Вибрати схему за варіантом з таблиці 2.1. Номер варіанту відповідає номеру студента у списку групи.

2.3.3) Намалювати умовне графічне зображення відповідної схеми і заповнити її таблицю істинності. Для варіантів з інверсними входами/виходами простіше спочатку заповнити пряму таблицю, а потім з інверсією.

2.3.4) Описати схему на мові VHDL за допомогою паралельних операторів умовного або вибіркового призначення у вигляді таблиці істинності, використовуючи векторну форму представлення сигналів типу bit_vector для багато розрядних сигналів.

2.3.5) Провести моделювання схеми з нульовими затримками. Для проведення моделювання на входи схеми необхідно подати такі послідовності вхідних сигналів, щоб перевірити її роботу за таблицею істинності. Дослідити результати моделювання.

Таблиця 2.1. Варіанти завдань до лабораторної роботи №2

Варіант	Логічна схема	Варіант	Логічна схема
1	DC з 2-розрядним 2-м кодом	20	DC з 2-розрядним 2-м кодом і inv входом
2	DC з 3-розрядним 2-м кодом	21	DC з 3-розрядним 2-м кодом і inv входом
3	CD з 2-розрядним 2-м кодом	22	CD з 2-розрядним 2-м кодом і inv входом
4	CD з 3-розрядним 2-м кодом	23	CD з 3-розрядним 2-м кодом і inv входом
5	DC з 2-розрядним 2-м кодом і inv виходом	24	DC з 2-розрядним 2-м кодом і inv виходом та входом
6	DC з 3-розрядним 2-м кодом і inv виходом	25	DC з 3-розрядним 2-м кодом і inv виходом та входом
7	CD з 2-розрядним 2-м кодом і inv виходом	26	CD з 2-розрядним 2-м кодом і inv виходом та входом
8	CD з 3-розрядним 2-м кодом і inv виходом	27	CD з 3-розрядним 2-м кодом і inv виходом та входом

Варіант	Логічна схема	Варіант	Логічна схема
9	MUX з 2-розрядною адресою	28	MUX з 2-розрядною inv адресою
10	DMX з 2-розрядною адресою	29	DMX з 2-розрядною inv адресою
11	MUX з 3-розрядною адресою	30	MUX з 3-розрядною inv адресою
12	DMX з 3-розрядною адресою	31	DMX з 3-розрядною inv адресою
13	MUX з 2-розрядною адресою і inv виходом	32	MUX з 2-розрядною inv адресою і inv виходом
14	DMX з 2-розрядною адресою і inv виходом	33	DMX з 2-розрядною inv адресою і inv виходом
15	MUX з 3-розрядною адресою і inv виходом	34	MUX з 3-розрядною inv адресою і inv виходом
16	DMX з 3-розрядною адресою і inv виходом	35	DMX з 3-розрядною inv адресою і inv виходом
17	Повний суматор	36	Повний суматор з inv виходами
18	Напівсуматор	37	Напівсуматор з inv виходом
19	Повний вираховувач	38	Повний вираховувач з inv виходами
39	Повний суматор з inv входами	40	Повний суматор з inv входами
41	Напівсуматор з inv входом	42	Напівсуматор з inv входом і виходом
43	Повний вираховувач з inv входами	44	Повний вираховувач з inv входами і виходами
45	Перевірка на парність на 3 розряди	46	Перевірка на парність на 3 розряди з inv виходом

Варіант	Логічна схема	Варіант	Логічна схема
47	Перевірка на непарність на 3 розряди	48	Перевірка на непарність на 3 розряди з inv виходом

2.4. Зміст звіту

2.4.1) Мета роботи.

2.4.2) Завдання за варіантом.

2.4.3) Умовне графічне зображення схеми і її таблиця істинності .

2.4.4) Текст програми і результати моделювання у табличній формі і у формі часових діаграм.

2.4.5) Висновки по роботі.

2.5. Контрольні запитання

2.5.1) Розповісти про принцип роботи своєї схеми.

2.5.2) Розповісти про основні типи сигналів у мові VHDL.

2.5.3) Що таке паралельні і послідовні оператори у мови VHDL?

2.5.4) Порівняльна характеристика поведінкових моделей, які підтримує мова VHDL: переваги, недоліки, область застосування.

ЛАБОРАТОРНА РОБОТА № 3
ПОБУДОВА СТРУКТУРНИХ МОДЕЛЕЙ ПОСЛІДОВНОСТНИХ СХЕМ НА МОВІ VHDL

3.1. Мета роботи

- побудова ієрархічних структурних моделей послідовних схем і перевірка правильності їх роботи;
- придбання досвіду використання сигналів типу std_logic, побудови структурного опису проекту з використанням констант типу Constant і Generic, паралельних операторів прямої реалізації інтерфейсу, створення компонентів і настроювання конфігурації у мові VHDL.

3.2. Рекомендації

Для виконання роботи необхідно повторити і ознайомитись з наступним теоретичним матеріалом:

- принципи роботи, схемна (вентильна) реалізація, таблиці істинності тригерів («Додаток В»);
- засоби створення структурних моделей цифрових схем у мові VHDL: скалярні сигнали типу std_logic, оператор безумовного призначення значення сигналу, константи типу Constant і Generic, побудова структурного опису проекту за допомогою паралельних операторів прямої реалізації інтерфейсу, створення екземплярів компонентів, створення модуля конфігурації у мові VHDL [1–4].
- Приклади моделей базових компонентів тригерів наведено нижче.

Приклад моделі логічного вентиля АБО-НІ з різними затримками вихідних сигналів при переключенні з 1→ 0 та 0→ 1:

```
library IEEE;
use IEEE.std_logic_1164.all;

entity nor2 is
   generic (rise_time: time := 5 ns;
            fall_time: time := 7 ns);
   port(I1 : in STD_LOGIC;
        I0 : in STD_LOGIC;
        OU : out STD_LOGIC);
end nor2;

architecture nor2 of nor2 is

begin
   with not(I0 or I1) select
   OU <= not(I0 or I1) after rise_time when '1',
      not(I0 or I1) after fall_time when others;
end nor2;
```

Приклад моделі асинхронного RS тригера з двома різними архітектурними тілами:

```
library IEEE;
   use IEEE.std_logic_1164.all;

   entity RS is
     generic (t_rise: time := 10 ns;
              t_fall: time := 15 ns);
    port(R : in STD_LOGIC;
         S : in STD_LOGIC;
         Q : inout STD_LOGIC;
         nQ : inout STD_LOGIC);
   end RS;
```

```
architecture RS_forward of RS is –пряма реалізація інтерфейсу
begin
  elem1: entity work. nor2
        generic map(rise_time => t_rise, fall_time => t_fall)
        port map (I1 => S, I0 => Q, OU => nQ);
  elem2: entity work. nor2
        generic map(rise_time => t_rise, fall_time => t_fall)
        port map (I1 => R, I0=> nQ, OU => Q);
end RS_forward;

architecture RS of RS is  – компонентна реалізація з різними видами
                --призначення параметрів екземплярам компонентів
component nor2
generic (rise_time: time := t_rise;
                    fall_time: time := t_fall);
port(I1 : in STD_LOGIC;
      I0 : in STD_LOGIC;
            OU: out STD_LOGIC);
end component;

begin
elem1: nor2
generic map(rise_time => 10 ns, fall_time => t_fall)
port map (B => S, C => Q, RES => nQ);

elem2: nor2
port map (B => R, C => nQ, RES => Q);
end RS;
```

3.3. Завдання

3.3.1) Ознайомитись з теоретичними відомостями.

3.3.2) Вибрати схему за варіантом з таблиці 3.1. Номер варіанту відповідає номеру студента у списку групи.

Таблиця 3.1. Варіанти завдань до лабораторної роботи №3

№ варіанта	Тип тригера	Синхронізація	Структура	Елементна база
1.	R	0	1-но ступеневий	Або-НІ
2.	S	0	1-но ступеневий	Або-НІ
3.	E	0	1-но ступеневий	Або-НІ
4.	$\overline{RS}$	0	1-но ступеневий	Або-НІ
5.	D	0	1-но ступеневий	Або-НІ
6.	R	1	1-но ступеневий	І-НІ
7.	S	1	1-но ступеневий	І-НІ
8.	RS	1	1-но ступеневий	І-НІ
9.	D	1	1-но ступеневий	І-НІ
10.	E	1	1-но ступеневий	І-НІ
11.	JK	фронт	1-но ступеневий	довільна
12.	T	фронт	1-но ступеневий	довільна
13.	RS	фронт	2-х ступеневий	І-НІ
14.	D	фронт	2-х ступеневий	Або-НІ

№ варіанта	Тип тригера	Синхронізація	Структура	Елементна база
15.	D	фронт	2-х ступеневий	І-НІ
16.	$\overline{RS}$	фронт	2-х ступеневий	Або-НІ
17.	$\overline{DRS}$	фронт	2-х ступеневий	І-НІ
18.	$\overline{TRS}$	фронт	2-х ступеневий	І-НІ
19.	$\overline{JKRS}$	фронт	2-х ступеневий	І-НІ
20.	$\overline{JKRS}$	фронт	2-х ступеневий	Або-НІ
21.	TRS	фронт	2-х ступеневий	Або-НІ
22.	DRS	фронт	2-х ступеневий	Або-НІ
23.	RS	фронт	6-вентильна	Або-НІ
24.	D	фронт	6-вентильна	І-НІ
25.	$\overline{TRS}$	фронт	6-вентильна	І-НІ
26.	$\overline{JKRS}$	фронт	6-вентильна	І-НІ
27.	$\overline{RS}$	фронт	6-вентильна	І-НІ
28.	D	фронт	6-вентильна	Або-НІ
29.	TRS	фронт	6-вентильна	Або-НІ
30.	$\overline{DRS}$	фронт	6-вентильна	І-НІ
31.	JKRS	фронт	6-вентильна	Або-НІ
32.	DRS	фронт	6-вентильна	Або-НІ
33.	D	фронт	з	І-НІ

№ варіанта	Тип тригера	Синхронізація	Структура	Елементна база
			забороненими зв’язками	
34.	–– TRS	фронт	з забороненими зв’язками	І-НІ
35.	–– JKRS	фронт	з забороненими зв’язками	І-НІ
36.	JKRS	фронт	з забороненими зв’язками	Або-НІ
37.	–– DRS	фронт	з забороненими зв’язками	І-НІ
38.	TRS	фронт	з забороненими зв’язками	Або-НІ
39.	D	фронт	з забороненими зв’язками	Або-НІ
40.	DRS	фронт	з забороненими зв’язками	Або-НІ

3.3.3) Намалювати вентильну схему тригеру і привести його таблицю істинності. На схемі необхідно позначити імена всіх входів, виходів, елементів і внутрішніх зв’язків.

3.3.4) Для кожного типу вентиля схеми створити окремий VHDL-файл і описати його архітектуру як булеве рівняння за допомогою паралельного оператора умовного призначення. Затримки сигналів на його

виході описати як константи типу Generic і призначити нулеві значення за замовчуванням. Для опису всіх сигналів використовувати скалярні сигнали типу std_logic. Бажаючі можуть написати алгоритмічну модель вентиля з використанням оператора Process і довільною кількістю входів.

3.3.5) Для комірки пам’яті (асинхронний RS тригер) створити окремий VHDL-файл і описати його архітектуру як структурну модель за допомогою прямої реалізації інтерфейсу.

3.3.6) За допомогою моделювання з нульовими затримками сигналів перевірити працездатність окремих моделей п.3.3.4. і п.3.3.5.

3.3.8) Створити VHDL-файл зі структурним описом схеми тригера, у якому базові елементи, моделі яких було створено раніше, об’явити як компоненти. Зв’язування екземплярів компонентів з моделями виконати за замовчуванням (імена компонентів і інтерфейсів моделей повинні співпадати).

3.3.9) Для проведення аналізу роботи схеми необхідно створити тестову послідовність вхідних сигналів, щоб перевірити її роботу:

- за типом синхронізації;

- за таблицею істинності;

- при переході між режимами роботи у різних комбінаціях (запис після зберігання, запис після запису і т.д.).

Для цього доцільно намалювати вхідні діаграми. ***Не слід забувати*** про необхідність передвстановлення і витримки інформаційних сигналів відносно синхроімпульсу. Для введення складних діаграм зручно використовувати гарячі кнопки, а потім запам’ятати діаграму як тип користувача.

3.3.10) Провести моделювання схеми з однаковими середніми затримками сигналів на виходах вентилів. Дослідити результати моделювання. Однакові затримки сигналів зручно використовувати при аналізі шляхів розповсюдження сигналів в схемі і пошуку помилок у її моделі. Для пошуку місця виникнення помилки потрібно вивести на діаграму проміжні сигнали в схемі і порівняти їх поточні значення з

очікуваними. Аналіз потрібно починати з моменту часу зміни вхідних сигналів, які призвели до невірного переключення виходів схеми.

3.3.13) Внести зміни у VHDL-файли базових елементів: змінити імена їх інтерфейсів або архітектур. Зв’язування екземплярів компонентів з моделями виконати за допомогою окремого файлу конфігурації. При цьому, модуль загальної схеми тригера буде зкомпільовано умовно, тобто із попередженнями, а зв’язування компонентів буде відбуватися під час моделювання.

3.3.14) Повторити моделювання на попередній послідовності вхідних сигналів. ***Звернути увагу*** на те, що у якості модуля верхнього рівня слід ставити саме модуль конфігурації.

3.4. Зміст звіту

3.4.1) Мета роботи.

3.4.2) Завдання за варіантом.

3.4.3) Таблиця істинності тригера і його вентильна схема з позначенням імен всіх елементів і сигналів.

3.4.4) Текст програми і результати моделювання у формі часових діаграм для пунктів 3.3.9, 3.3.10, 3.3.12, 3.13.4.

3.4.5) Висновки по роботі, зокрема, особливості у роботі даної схемної реалізації тригера.

3.5. Контрольні запитання

3.5.1) Розповісти про принцип роботи своєї схеми і особливості її роботи.

3.5.2) Розповісти про структурні елементи проекту на прикладі своєї моделі.

3.5.3) Що таке Constant і Generic у мови VHDL і чим вони відрізняються?

3.5.4) Що таке оператор прямої реалізації інтерфейсу і оператор створення екземпляру компоненту у мова VHDL (переваги, недоліки, область застосування)?

3.5.5) Яке призначення модуля конфігурації у мові VHDL?

3.5.6) При яких умовах при моделюванні виникає помилка генерації сигналів?

ЛАБОРАТОРНА РОБОТА № 4
АНАЛІЗ ЧАСОВИХ ПАРАМЕТРІВ ПОСЛІДОВНОСТНИХ СХЕМ З ВИКОРИСТАННЯМ ЗАСОБІВ МОВИ VHDL

4.1. Мета роботи

- побудова динамічний моделей послідовностних схем і визначення їх основних динамічних параметрів: максимально припустима частота синхронізації, час передвстановлення та витримки інформаційних сигналів відносно синхросигналів, мінімально припустима довжина вхідних сигналів;
- придбання досвіду аналізу часових діаграм роботи пристрою;
- придбання досвіду опису динамічних параметрів елементів схеми за допомогою операторів transport, after, reject і констант типу Constant і Generic мови VHDL.

4.2. Рекомендації

Для виконання роботи необхідно повторити і ознайомитись з наступним теоретичним матеріалом:

- динамічні параметри тригерних схем та приклади їх визначення (Додаток Г);
- засоби опису динамічних параметрів елементів схеми за допомогою операторів transport, after, reject і констант типу Constant і Generic у мові VHDL [1–4].

До основних часових параметрів тригерів можна віднести:

- мінімальна і максимальна затримка тригера;
- час передвстановлення та витримки інформаційних сигналів відносно сигналів синхронізації;
- мінімально припустима тривалість інформаційних сигналів і сигналів асинхронного RS- встановлення;

- мінімально припустима довжина такту синхронізації і максимально припустима частота синхронізації.

Параметри вимірюються по часовій діаграмі роботи тригера в різних режимах його роботи.

4.3. Завдання

4.3.1) Ознайомтесь з теоретичними відомостями.

4.3.2) Задайте всім вентилям схеми з лабораторної роботи №3 затримки, що відрізняються для переднього і заднього фронту сигналів, наприклад, у такому співвідношенні: $t_{з01} = 0.7t_{з10}$.

4.3.3) Виконайте моделювання схеми з лабораторної роботи №3 на попередній послідовності вхідних сигналів для одного з варіантів реалізації моделі схеми.

4.3.4) Проведіть вимірювання і розрахунки динамічних параметрів тригера: мінімальної і максимальної затримка тригера; мінімально припустимої тривалості інформаційних сигналів і сигналів асинхронного RS- встановлення; часу передвстановлення та витримки інформаційних сигналів відносно сигналів синхронізації; мінімально припустимої довжини такту синхронізації і мінімально припустимої частоти синхронізації.

4.4. Зміст звіту

4.1) Мета роботи.

4.2) Завдання.

4.3) Структурна схема ЦС.

4.4) Тексти програм і результати моделювання у асинхроному режимі.

4.5) Розрахунки динамічних параметрів ЦС на основі діаграм її роботи.

4.6) Висновки по роботі.

4.5. Контрольні запитання

5.1) Розповісти про алгоритм роботи і таблицю істинності свого пристрою.

5.2) Розповісти про побудову моделі свого пристрою на мові VHDL.

5.3) Що таке Constant і Generic у мови VHDL і чим вони відрізняються?

5.5) До яких наслідків може привести недотримання мінімально допустимих значень динамічних параметрів?

5.6) Чим відрізняються оператори transport, after, reject у мова VHDL (переваги, недоліки, область застосування)?

ЛАБОРАТОРНА РОБОТА № 5
ПОБУДОВА АЛГОРИТМІЧНИХ МОДЕЛЕЙ ПОСЛІДОВНОСТНИХ СХЕМ НА МОВІ VHDL

5.1. Мета роботи

- побудова алгоритмічних моделей послідовних схем;
- вивчення принципів організації подієвого моделювання і особливостей створення поведінкових моделей паралельних процесів;
- придбання досвіду опису цифрових пристроїв на функціональному рівні абстракції з використанням паралельного оператора процесу, послідовних операторів, процедур та функцій мови VHDL.

5.2. Рекомендації

Для виконання роботи необхідно повторити і ознайомитись з наступним теоретичним матеріалом:

- основними відомостями про регістри і лічильники («Додаток Д»);
- алгоритмом подієвого моделювання, поняттям списку чутливих входів процесу і списку зв’язків сигналів, з атрибутами сигналів і типом сигналів std_logic_vector [1–4];
- засобами створення поведінкових моделей у мові VHDL: операторами Process, Wait, if, case, loop, процедурами і функціями, стандартними бібліотеками арифметичних операцій [1–4].
- Приклади опису арифметичних операцій в моделях ЦП надано в Додатку Е.
- Для створення алгоритмічної моделі пристрою функціонального рівня треба уявляти принцип його побудови і особливості роботи.

- Рекомендації по написанню моделей за варіантами завдань

Алгоритмічну модель типового блоку зручно написана на основі його ***узагальненої таблиці істинності*** (приклад в Додатку Д - багатофункціональні регістри і лічильники). При складанні таблиці істинності треба, насамперед, правильно визначити ***пріоритети входів***. А для цього потрібно добре уявляти загальну побудову і принцип роботи схеми. Багатофункціональні послідовні пристрої у загальному вигляді складаються з розрядних схем, побудованих на синхронних тригерах з додаванням комбінаційної логіки. Наприклад, у схемах, що базуються на тригерах з динамічним управлінням, найчастіше самими пріоритетними є входи асинхронного встановлення R та S, потім – вхід синхронізації C, потім входи вибору режиму роботи, і в останню чергу – інформаційні входи. Але можливі і інші варіанти, наприклад, коли входи вибору режиму роботи можуть блокувати дію входу синхронізації, або коли паралельне завантаження вхідного коду відбувається асинхронно.

У пристроях, які можуть керувати формою представлення вихідної інформації, сама інформація зберігається у тригерах, до виходів яких під'єднані керовані комбінаційні схеми. Наприклад, регістри з тристабільними виходами, або з прямим чи інверсним кодом на виходах. В такому випадку з точки зору мови VHDL процес запису та зберігання інформації у тригерах і процес перетворення вихідних даних є незалежними процесами. Їх зручніше описувати декількома операторами Process.

Якщо модель вузла створюється для перевірки правильності роботи проекту, у якому цей вузол буде використовуватись в якості елементу, то модель повинна максимально виявляти можливі помилки розробника. Вона повинна адекватно реагувати на всі значеннях вхідних сигналів: 0,1,X,U,Z і т.д., і не вводити розробника в оману, наприклад, не призначати значення 0 вихідним сигналам, коли сигнали на впливових входах приймають не передбачені алгоритмом роботи значення. У випадках, коли не можливо передбачити, якою буде реакція вузла на певні комбінації

вхідних сигналів (наприклад невизначені значення сигналів на управляючих входах), потрібно призначати «X» вихідним сигналам.

Для зручності написання моделі у 9-значній логіці краще максимально використовувати ***стандартні операції і рівняння*** (вони вже реалізовані у багатозначній логіці). В якості аргументів у виразах використовувати ***сигнали та змінні*** і по можливості запобігати прямого перебору константних значень в умовних операторах. В наведеному нижче прикладі моделі JK- тригеру для визначення значення вихідного сигналу простіше скористатися характеристичним рівнянням тригера, ніж перебирати всі значення J і K за допомогою умовних операторів. А, ось коректна типова конструкція прийому вхідних даних:

```
if D(i) =’0’|’1’ then Q(i)<=D(i);
else Q(i)<=’X’;
 end if;
```

Але, якщо просто написати Q(i)<=D(i); то це може бути навіть більш інформативно. Типовий приклад – певний регістр зчитує дані з загальної три стабільної шини. У випадку помилки керування в момент зчитування даних на входах регістру може з’явитись високий імпеданс ‘Z’. Модель регістру одразу покаже цю ситуацію на його виходах. Тоді як в першому варіанті виходам моделі буде призначено ’X’. Це може свідчити про різні некоректні ситуації в роботі регістру, і потребує додаткового аналізу.

Опис алгоритмів роботи багатофункціональних пристроїв передбачає перевірки ряду умов вибору режиму його роботи в залежності від комбінацій вхідних сигналів. Типовою помилкою, яку можна зустріти на цьому етапі, є невірне врахування альтернативних гілок алгоритму. Найчастіше це трапляється при застосуванні комплексних умов у wait, case операторах або при застосуванні багато вкладених if конструкцій. Наприклад,

```
Signal R,C,PL: bit;
process (R,C)
begin
if ((R=’0’) and (rising_edge (C))) then
```

```
        if PL='1' then <режим1>;
        else <режим2>;
        end if;
end if;
end process;
```

Тобто, при зміні сигналів R або C перевіряється умова ((R='0') and (rising_edge (C)). При її виконанні в залежності від значення PL вибирається режими 1 або 2. При невиконанні умови – в процесі нічого не відбувається (стан зберігання). Тоді, як насправді, це буде вірно тільки при умові (R='0') а при R='1' – ні.

В таб. 5.1 приведено приклад заповнення таблиці істинності в багатозначній логіці для JK- тригера с RS-встановленням та синхронізацією за переднім фронтом C.

Таблиця 5.1. Таблиця істинності в багатозначній логіці

Режим	**Входи**				**Виходи**	
	R	**S**	**C**	**J,K**	**Q**	**$\overline{Q}$**
Скидання	**1**	**0**	**–**	**–**	**0**	**1**
Встановлення	**0**	**1**	**–**	**–**	**1**	**0**
Заборонено	**1**	**1**	**–**	**–**	**X**	**X**
Невизначено	**X, Z, U,….**		**–**	**–**	**X**	**X**
Режим JK	**0**	**0**	**↑**	**0,1**	**F(J,K,Q')**	
Зберігання	**0**	**0**	**↓,0,1**	**–**	**Q'**	**$\overline{Q}$'**
Невизначено	**0**	**0**	**X, Z, U,….**	**–**	**X**	**X**
Невизначено	**0**	**0**	**↑**	**X, Z, U,….**	**X**	**X**

Таблиця складається за порядком пріоритетності входів, які визначають режими роботи елементу: R та S; C; J та K.

“–“ означає, що сигнал на вході не впливає на роботу схеми в даному режимі. Наприклад, в режимі асинхронного встановлення сигнали на C, J, K не мають значення.

Коли на впливовому (в певному режимі) вході з’являються не передбачені логікою роботи значення сигналів, модель не може адекватно визначити результат. В такому випадку значення X на виходах моделі попереджає про це розробника, який буде використовувати модель елементу в своїй схемі.

В моделі замість стандартної функції визначення переднього фронту *rising_edge(C)* використовується функція *posedge (C)*. Стандартна функція повертає TRUE, якщо C зміниться з 0→1, а у всіх інших випадках повертає FALSE, що при використанні функції буде відповідати режиму зберігання. Функція *posedge (C)* – повертає три значення (1,0,X) що дає змогу розрізняти рижим зберігання і наявність некоректного сигналу на вході C.

Приклад алгоритмічної моделі послідовностної схеми:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity lab5JK is
Port(
C,R,S,J,K: in STD_LOGIC;
Q,nQ: inout STD_LOGIC);
end lab5JK;

architecture lab5JK of lab5JK is
------------------ функції-------------------------
-- не стандартна функція визначення переднього фронту для тестування: 1- є фронт,
-- 0 – немає фронту (зберігання), X - не можна визначити, чи був фронт
function posedge (signal s: STD_LOGIC)
return STD_LOGIC is
begin
if s'event=FALSE then return '0';
        elsif s'last_value ='0' and s ='1' then return '1';
```

```
                elsif s='0' then return '0';
                        else return 'X';
end if;
end posedge;
        ------------------------------------------------
begin
process
variable vq: STD_LOGIC;
begin
        wait on R,S,C;
if (R='1'and S='0') then vq:='0';
        elsif (R='0'and S='1') then vq:='1';
        elsif (R='0'and S='0') then
                if posedge(C)= '1' then vq:=(nQ and J)or(Q and not( K));
            elsif posedge(C)= '0' then null;
                else vq:='X';
            end if;
else vq:='X';
end if;
Q<= vq;
nQ<= not(vq);
end process;
end lab5;
```

5.3. Завдання

5.3.1) Ознайомитись з теоретичними відомостями.

5.3.2) Вибрати тип пристрою за варіантом з таблиці 5.1. Номер варіанту відповідає номеру студента у списку групи. Кількість розрядів схеми задавати через параметр (default = 4).

Таблиця 5.1. Варіанти завдань до лабораторної роботи № 5

№ варіанта	Тип пристрою	Синхронізація	Додаткові входи
1.	Послідовно -	фронт	з $\overline{R}$ -

№ варіанта	Тип пристрою	Синхронізація	Додаткові входи
	паралельний РГ (<)		встановленням
2.	Послідовно - паралельний РГ (>)	фронт	з R-встановленням
3.	Послідовно - паралельний РГ (<, >)	фронт	з R-встановленням
4.	Послідовно - паралельний РГ (<)	фронт	з R-встановленням і З-х стабільними виходами
5.	Послідовно - паралельний РГ (>)	фронт	з R-встановленням і З-х стабільними виходами
6.	Послідовно - паралельний РГ (<, >)	фронт	з R-встановленням і З-х стабільними виходами
7.	Лічильник (+)	фронт	з $\overline{R}$ - встановленням
8.	Лічильник (-)	фронт	з R-встановленням
9.	Лічильник (+, -)	фронт	з R-встановленням
10.	Лічильник (+)	фронт	з паралельним завантаженням
11.	Лічильник (-)	фронт	з паралельним завантаженням
12.	Послідовно - паралельний РГ (<)	фронт	з паралельним завантаженням

№ варіанта	Тип пристрою	Синхронізація	Додаткові входи
13.	Послідовно - паралельний РГ (>)	фронт	з паралельним завантаженням
14.	Послідовно - паралельний РГ (<, >)	фронт	з паралельним завантаженням
15.	Послідовний РГ (<)	фронт	з R-встановленням
16.	Послідовний РГ (>)	фронт	з $\overline{R}$ - встановленням
17.	Послідовний РГ (<, >)	фронт	з R-встановленням
18.	Послідовний РГ (<)	фронт	з паралельним завантаженням
19.	Послідовний РГ (>)	фронт	з паралельним завантаженням
20.	Послідовний РГ (< >)	фронт	з паралельним завантаженням
21.	Суматор з накопиченням результату	фронт	з R-встановленням
22.	Паралельний РГ	фронт	з $\overline{R}$ - встановленням
23.	Паралельний РГ	фронт	з 3-х стабільними виходами
24.	Суматор з накопиченням результату	фронт	з паралельним завантаженням
25.	Лічильник (+ -)	фронт	з паралельним завантаженням

№ варіанта	Тип пристрою	Синхронізація	Додаткові входи
26.	Вирахувач з накопиченням результату	фронт	з паралельним завантаженням
27.	Паралельний РГ	1	з вихідним кодом у прямому і інверсному коді
28.	Паралельний РГ	фронт	з вихідним кодом у прямому і інверсному коді
29.	Паралельний РГ	0	з 3-х стабільними виходами
30.	Паралельний РГ	фронт	з 3-х стабільними виходами і з $\overline{R}$ - встановленням
31.	Послідовно - паралельний РГ (<)	фронт	з $\overline{R}$ - встановленням і паралел. завантаженням
32.	Послідовно - паралельний кільцевий РГ (>)	фронт	з вибором: послідовний вхід або кільце
33.	Послідовно - паралельний кільцевий РГ (<)	фронт	з вибором: послідовний вхід або кільце
34.	Схема множення 2-х чисел зі збереженням результату	фронт	з R-встановленням
35.	Схема множення числа на 3 зі збереженням	фронт	з $\overline{R}$ - встановленням

№ варіанта	Тип пристрою	Синхронізація	Додаткові входи
	результату		
36.	Кільцевий послідовно - паралельний РГ (<)	фронт	з паралельним завантаженням
37.	Кільцевий послідовно - паралельний РГ (>)	фронт	з паралельним завантаженням
38.	Схема віднімання 2-х чисел зі збереженням результату	фронт	з R-встановленням
39.	Схема складання 2-х чисел зі збереженням результату	фронт	з R-встановленням
40.	Послідовно - паралельний РГ (>)	фронт	з R-встановленням і паралел. завантаженням

5.3.3) Для пояснення принципу роботи заданого пристрою навести його структурну схему та скласти відповідну таблицю істинності.

5.3.4) Функціонування пристрою описати за допомогою одного чи декількох паралельних операторів процесу. Для реалізації арифметичних операцій використати функції зі стандартних бібліотек або власні процедури і функції (для операцій складання, інкременту, тощо). Для опису багато розрядних сигналів використати тип std_logic_vector. Передбачити адекватну реакцію моделі на значення вхідних сигналів X,U,Z та ін., а не тільки на 0 і 1.

5.3.5) Для проведення аналізу роботи схеми необхідно створити тестову послідовність вхідних сигналів, яка ***перевіряє всі режими її роботи*** за таблицею істинності.

Для цього доцільно спочатку намалювати вхідні діаграми. Для введення складних діаграм за допомогою графічного редактора діаграм зручно використовувати гарячі кнопки, а потім запам’ятати діаграму як тип користувача.

5.3.6) Провести моделювання ЦП з нульовими затримками сигналів. Дослідити результати моделювання.

5.4. Зміст звіту

5.4.1) Мета роботи.

5.4.2) Завдання.

5.4.3) Структурна схема і таблиця істинності ЦС.

5.4.4) Текст програми (з коментарями у електронному вигляді здається викладачу) і результати моделювання .

5.4.5) Висновки по роботі.

5.5. Контрольні запитання

5.5.1) Розповісти про принцип роботи свого пристрою і показати відповідність йому складеної таблиці істинності.

5.5.2) Що таке список чутливих входів процесу і список зв’язків сигналів, які атрибути сигналів пов’язані з процесом моделювання у мові VHDL?

5.5.3) Які типи атрибутів сигналів мови VHDL вам відомі?

5.5.4) Які значення може приймати сигнал/змінна типу std_logic і, що вони означають?

5.5.5) Чим відрізняються алгоритмічні моделі від інших типів?

5.5.6) До якого типу (паралельні, послідовні) відносяться оператори мови VHDL: Process, Wait, if, case, loop, виклик процедур і функцій?

5.5.7) Як реалізувати арифметичні операції засобами мови VHDL?

ЛАБОРАТОРНА РОБОТА № 6
АНАЛІЗ РОБОТИ ПОСЛІДОВНОСТНИХ СХЕМ З ВИКОРИСТАННЯМ ДИНАМІЧНИХ АЛГОРИТМІЧНИХ МОДЕЛЕЙ У МОВІ VHDL

6.1. Мета роботи

- побудова динамічних алгоритмічних моделей послідовних схем;
- побудова тестової послідовності вхідних сигналів Testbanch з перевіркою співвідношень вхідних сигналів;
- придбання досвіду опису затримок вихідних сигналів цифрового пристрою в залежності від режиму його роботи.

6.2. Рекомендації

Для виконання роботи необхідно повторити і ознайомитись з наступним теоретичним матеріалом:

- динамічні параметри послідовностних схем: затримки шляхів сигналів від різних входів до різних виходів; час передвстановлення та витримки інформаційних сигналів відносно сигналів синхронізації; мінімально припустима тривалість вхідних сигналів (див. лабораторну роботу №4).
- засоби опису динамічних параметрів елементів схеми за допомогою операторів transport, after, reject та констант в операторі Process; оператор повідомлення assert в мові VHDL [1–4].
- засобами створення Testbanch у вигляді процесу у мови VHDL: операторами Process, Wait, if, case, loop [1–4].
- Рекомендації по написанню динамічних моделей і Testbanch.

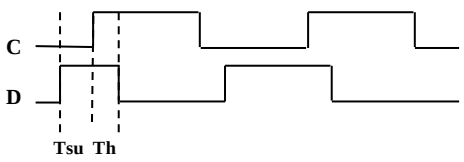
Для створення Testbanch у вигляді процесу треба використати діаграми вхідних сигналів з лабораторної робот №5, що створювались для перевірки **всіх режимів** роботи моделі за таблицею істинності з

урахуванням в 9-ти значній логіці. Всі часові інтервали між зміною вхідних сигналів треба збільшити, враховуючи не нульові значення затримок у моделі пристрою.

Для створення динамічної алгоритмічної моделі пристрою треба знати принцип його побудови і особливості роботи. На основі узагальненої таблиці істинності пристрою необхідно визначити, від зміни яких саме вхідних сигналів змінюються вихідні сигнали у кожному режимі роботи пристрою. Не слід забувати призначати певні значення затримок і при переключенні виходів моделі у невизначений стан. Наприклад, призначати найгірше значення (найбільше або найменше) з можливих у конкретному випадку. Можна ввести **змінні для затримок**, значення яких будуть рахуватись всередині оператора Process (при описі режимів роботи пристрою), а в кінці – привласнюватись вихідним сигналам.

Приведемо приклад моделі 4-х розрядного регістру зсуву з R встановленням і синхронізацією за переднім фронтом Clock. В моделі використовується функція визначення переднього фронту сигналу, описана в лаб.5 і процедура організації зсуву в регістрі, яка не розкрита з метою самостійності виконання відповідних варіантів завдань. Відмітимо тільки, що для визначення значень молодшого, старшого розряду і довжини вектора A використовуються його атрибути: A'low, A'high, A'length. Даний регістр може працювати у 2-х режимах: асинхронного скидання та зсуву вхідних даних. В цих режимах його вихідні сигнали Qi будуть перемикатись від різних вхідних сигналів з різними затримками.

У TestBench введено перевірку дотримання часових параметрів перед встановлення і витримки сигналу D відносно сигналу C (рис.6.1, докладно розглядались в лаб.4) з видачею попереджувального повідомлення.



Tsu (setup) і Th (hold) – константи типу time – час передвстановлення та витримки інформаційного сигналу D відносно дозволяючого фронту сигналу C (на рисунку переднього).

Рис. 6.1. Час передвстановлення (Tsu) та витримки (Th)

Динамічна модель регістру зсуву:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity shift_reg is
generic (size : integer := 4; -- розрядність регистру
        reset_delay : time := 0ns; -- затримка виходів регістру від входу R в режимі
скидання
        data_delay : time := 0ns); -- затримка виходів регістру від входу C в режимі зсуву

        port(D: in std_logic; -- вихід послідовного введення даних
        C: in std_logic; -- вихід синхронізації
        R: in std_logic; -- вихід асинхронного скидання
        Q: inout std_logic_vector (size - 1 downto 0)); -- виходи регістру
end shift_reg;

architecture shift_reg_arch of shift_reg is
---------- процедури, функції----------
function posedge (signal s: std_logic)
return std_logic is
begin
< тіло фінкції>
end posedge;

procedure right_shift(
signal A: inout std_logic_vector;     -- виходи тригерів регістру
signal D: in std_logic;               -- вхід послідовного введення даних
constant delay : time) is-- затримка виходів регістру від входу C в режимі зсуву
begin
< тіло процедури>
end right_shift;
        ----------------------------------------------------
begin
process
begin
        wait on C, R;
        if R = '1' then
                Q <= (others => '0') after reset_delay;
        elsif R = '0' then
                if posedge (C) = '1' then
                        right_shift (Q, D, data_delay);
```

```
                elsif posedge (C) = '0' then null;
            else Q <= (others => 'X') after data_delay;
            end if;
        else
        Q <= (others => 'X') after reset_delay;
        end if;
end process;
end shift_reg_arch;
```

Тестова послідовність вхідних сигналів:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity shift_reg_test is
end shift_reg_test;

architecture shift_reg_test of shift_reg_test is
component shift_reg is
generic (size : integer := 4;
reset_delay : time;
data_delay : time);
port( D: in std_logic;
C: in std_logic;
R: in std_logic;
Q: inout std_logic_vector(size - 1 downto 0));
end component;

constant reg_size : integer := 4;
constant clock_period : time := 100 ns;
constant t_setup : time := 30 ns;
constant t_hold : time := 10 ns;
signal D, C, R : std_logic;
signal Q : std_logic_vector(reg_size - 1 downto 0);

begin
reg: shift_reg
generic map (reset_delay => 20ns, data_delay => 40ns)
port map (D => D, C => C, R => R, Q => Q);
clock: process
begin
for i in 44 downto 0 loop
```

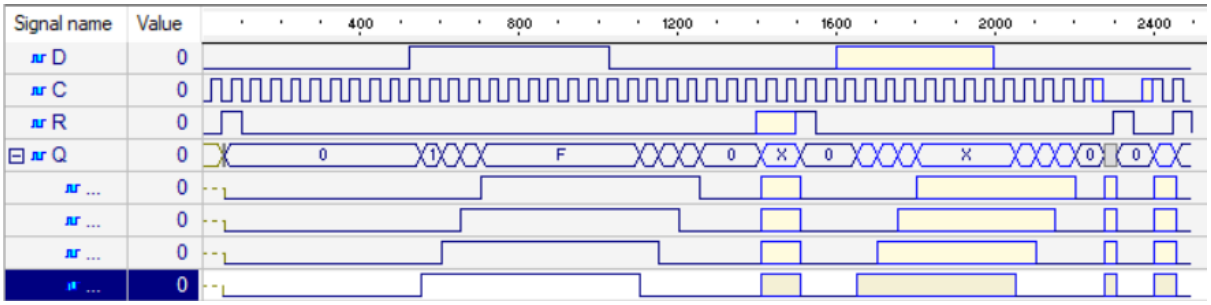
```
C <= '0';
wait for clock_period / 2;
C <= '1';
wait for clock_period / 2;
end loop;
C <= 'X';
wait for clock_period / 2;
C <= '0';
wait for clock_period * 2;
C <= 'X';
wait for clock_period / 2;
for i in 1 downto 0 loop
C <= '1';
wait for clock_period / 2;
C <= '0';
wait for clock_period / 2;
end loop;
end process;

reset: process
begin
R <= '0';
wait for clock_period;
R <= '1';
wait for clock_period;
R <= '0';
wait for 26 * clock_period;
R <= 'X';
wait for 2 * clock_period;
R <= '1';
wait for clock_period;
R <= '0';
wait for 15 * clock_period;
R <= '1';
wait for clock_period;
R <= '0';
wait for 2 * clock_period;
R <= '1';
wait for clock_period;
end process;
```

```
data: process
begin
D <= '0';
wait for 10.5 * clock_period;
D <= '1';
wait for 10.1 * clock_period;
D <= '0';
wait for 11.4 * clock_period;
D <= 'X';
wait for 8 * clock_period;
D <= '0';
wait for 10 * clock_period;
end process;

setup_hold_check: process
begin
wait until rising_edge(C);
assert(D'stable(t_setup))
report("Wrong setup time!")
warning error;
wait for t_hold;
assert(D'stable(t_hold))
report("Wrong hold time!")
warning error;
end process;
end shift_reg_test;
```

Результати моделювання:



6.3. Завдання

6.3.1) Ознайомитись з теоретичними відомостями.

6.3.2) Створити тестову послідовність вхідних сигналів Testbanch у вигляді процесу на основі тестових діаграм вхідних сигналів з лабораторної роботи №5.

6.3.3) Провести моделювання Testbanch, щоб упевнитися у відповідності результатів її роботи до бажаних часових діаграм.

6.3.4) Призначити різні значення затримок виходам пристрою з лабораторної роботи №5 у залежності від режимів його роботи. При виборі значень затримок можна орієнтуватись на значення динамічних параметрів тригерів з лабораторної роботи № 4.

6.3.5) Провести динамічне моделювання пристрою (з затримками сигналів) з використанням Testbanch.

6.3.6) Ввести оператор повідомлення, за допомогою якого перевірялися б певні співвідношення вхідних сигналів, наприклад, час передвстановлення та витримки інформаційних сигналів відносно сигналів синхронізації, мінімально припустима тривалість вхідних сигналів.

6.3.7) Дослідити результати моделювання.

6.4. Зміст звіту

6.4.1) Мета роботи.

6.4.2) Завдання.

6.4.3) Структурна схема і таблиця істинності ЦС з лабораторної роботи № 5.

6.4.4) Текст програми (з коментарями у електронному вигляді здається викладачу) і результати моделювання .

6.4.5) Висновки по роботі.

6.5. Контрольні запитання

6.5.1) Розповісти про принцип привласнення значень затримок у моделі свого пристрою.

6.5.2) Розповісти про структуру створеної Testbanch.

6.5.3) З якою метою створюють Testbanch у вигляді програми?

6.5.4) За яких умов виконується оператор повідомлення у мови VHDL?

6.5.5) Що можна перевірити за допомогою операторів повідомлення мови VHDL?

6.5.6) До якого типу (паралельні, послідовні) відносяться оператор повідомлення?

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Семенец В.В, Хаханова В.І., Хаханов В.І. Проектування цифрових систем з використанням мови VHDL. Навчальний посібник. – Харків: ХНУРЕ, 2003.-492 с.

2. Сергієнко А.М. VHDL для проектування обчислювальних пристроїв.–К.: ПП «Корнійчук» ТОВ. «ТИД «ДС», 2003. -208с.

3. Brock J. LaMeres. Introduction to Logic Ciurcits & Logic Design with VHDL, 2nd Edition. – Springer, 2019. – 489 pp.

4. Yalamanchili S. Introductory VHDL: From Simulation to Synthesis. Prentice-Hall, 2001. – 401 pp.

ДОДАТКОВА ЛІТЕРАТУРА

5. Бабич М.П., Жуков І.А. Комп’ютерна схемотехніка. Навчальний посібник. –К.: «МК-Прес», 2004. – 412с.

6. Справочник по цифровой схемотехнике. В.И.Зубчук, В.П.Сигорский, А.Н.Шкуро. –К.: Техника,1990. –105с.

7. Інтернет сайт фірми Aldec Inc.: http://www.aldec.com.

8. Інтернет сайт фірми Xilinx: http://www.xilinx.com

ДОДАТОК А.
БУЛЕВІ ФУНКЦІЇ

Булеві функції Fi однієї чи двох змінних (при n=1,2) мають назву елементарних. Загалом існує 4 елементарні булеві функції однієї змінної і 16 елементарних булевих функцій двох змінних. Відповідні таблиці істинності наведено нижче (табл. А.1 та А.2). Систему булевих функцій Fk, k=0,15, вважають за функціонально повну, якщо на її основі можна отримати будь-яку булеву функцію з наборів {f0..f3, F0..F15}. Якщо при видаленні з системи хоча б однієї функції, вона стає неповною, то така система називається мінімально повною. Набори логічних елементів, які реалізують функціонально повну, не обов'язково мінімальну, систему булевих функцій називаються базисом. Наприклад, можуть бути такі базиси:

1. x & y – І; x v y – АБО; $\bar{x}$ – НІ.
2. ($\overline{x \& y}$) – І-НІ (Штрих Шиффера).
3. ($\overline{x \vee y}$) – АБО-НІ (Стрілка Пірса).
4. ($\overline{x \& y}$) ; x & y.
5. x v y ; x & y; ($\overline{x \& y}$).

Елементарні болеві функції розширюються до n змінних та реалізуються інтегральними логічними елементами із більшою кількістю входів (n>2).

Тотожності булевої алгебри:

1. A&B=B&A комутативний закон
2. AvB=BvA комутативний закон
3. (AvB)vC=Av(BvC) асоціативний закон
4. (A&B)&C=A&(B&C) асоціативний закон
5. A&(BvC)=A&BvB&C дистрибутивний закон
6. AvB&C=(AvB)&(AvC) дистрибутивний закон

7. $A \vee \overline{A} = 1$
8. $A \& \overline{A} = 0$
9. $A \vee A = A$
10. $A \& A = A$
11. $\overline{(\overline{A})} = A$
12. $A \vee 1 = 1$
13. $A \& 0 = 0$
14. $\overline{(A \vee B)} = \overline{A} \& \overline{B}$ теорема де Моргана
15. $\overline{(A \& B)} = \overline{A} \vee \overline{B}$ теорема де Моргана

Таблиця А.1. Елементарні булеві функції однієї змінної

Функція	**Аргумент x функції / Результат fi**		**Позначення**	**Назва функції**
x	0	1		
f0	0	0	0	Константа 0
f1	1	1	1	Константа 1
f2	0	1	x	Змінна x
f3	1	0	$\bar{x}$	Заперечення x

Таблиця А.2. Елементарні булеві функції двох змінних

Функція	**Аргумент x функції / Результат Fi**				**Позначення**	**Назва функції**
x y	0 0	0 1	1 0	1 1		
F0	0	0	0	0	0	Константа 0
F1	0	0	0	1	xy	Логічне І
F2	0	0	1	0	$x(\bar{y})$	Заборона y
F3	0	0	1	1	x	Змінна x
F4	0	1	0	0	$(\bar{x})y$	Заборона x
F5	0	1	0	1	y	Змінна y
f6	0	1	1	0	x mod y	Сума за модулем 2

Функція	**Аргумент x функції / Результат Fi**				**Позначення**	**Назва функції**
f7	0	1	1	1	x v y	Логічне АБО
f8	1	0	0	0	$(\overline{x \vee y})$	Стрілка Пірса
f9	1	0	0	1	$(\overline{x \bmod 2\ y})$	Еквівалентність
f10	1	0	1	0	$\bar{y}$	Заперечення y
f11	1	0	1	1	y->x	Імплікація y від -x
f12	1	1	0	0	$\bar{x}$	Заперечення x
f13	1	1	0	1	x->y	Імплікація x від -y
f14	0	1	0	1	$(\overline{x\&y})$	Штрих Шиффера
f15	1	1	1	1	1	Константа 1

ДОДАТОК Б.
КОРОТКІ ВІДОМОСТІ ПРО ШИФРАТОРИ І ДЕШИФРАТОРИ, МУЛЬТИПЛЕКСОРИ І ДЕМУЛЬТИПЛЕКСОРИ

Б.1. ДЕШИФРАТОРИ І ШИФРАТОРИ

Дешифратори і шифратори – це комбінаційні пристрої, які належать до перетворювачів кодів.

Шифратор – це пристрій, який при збудженні одного зі своїх входів формує на виходах двійковий код, що відповідає номеру збудженої вхідної лінії, тобто перетворює код “1 з N” у двійковий код. Шифратор, що має 2^n входів та n виходів називається *повним*. Якщо число входів шифратора менше ніж 2^n, він називається *не повним*.

Шифратори широко використовуються для перетворення десяткових цифр і буквених символів на двійковий код при введенні інформації (натисненні відповідної клавіші пристрою введення) в ЕОМ та інших цифрових пристроях.

Таблицю істинності повного двійкового шифратора (перетворює код “1 з N” у двійковий код 8421) наведено у таб. Б.1. Умовне графічне позначення шифратора показано на рис. Б.1а.

Таблиця Б.1. Таблиця істинності двійкового шифратора

Входи				Виходи	
x3	x2	x1	x0	y1	y0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1
0	0	0	0	0	0

Дешифратор – пристрій, який виконує перетворення двійкового коду в код «1 из N», тобто у відповідь на певний двійковий код на своєму вході видає активний сигнал (наприклад, “1”) на вихід з відповідним номером, на решті виходів залишаються пасивні значення сигналів (наприклад, “0”). Таким чином, дешифратор виконує операцію, обернену до шифратора.

У загальному випадку дешифратор з n входами має 2^n виходів, оскільки n-розрядний код вхідного слова може приймати 2^n різних значень і кожному з цих значень відповідає сигнал 1 на одному з виходів дешифратора. Якщо частина вхідних наборів не використовується, то дешифратор називається *неповним* і у нього Nвих < 2^n.

Таблицю істинності повного двійкового дешифратора (перетворює двійковий код 8421 у код “1 з N”) наведено у таб. Б.2. Умовне графічне позначення шифратора показано на рис. Б.1б.

Таблиця Б.2. Таблиця істинності двійкового дешифратора

Входи		Виходи			
*x*1	*x*0	*y*3	*y*2	*y*1	*y*0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

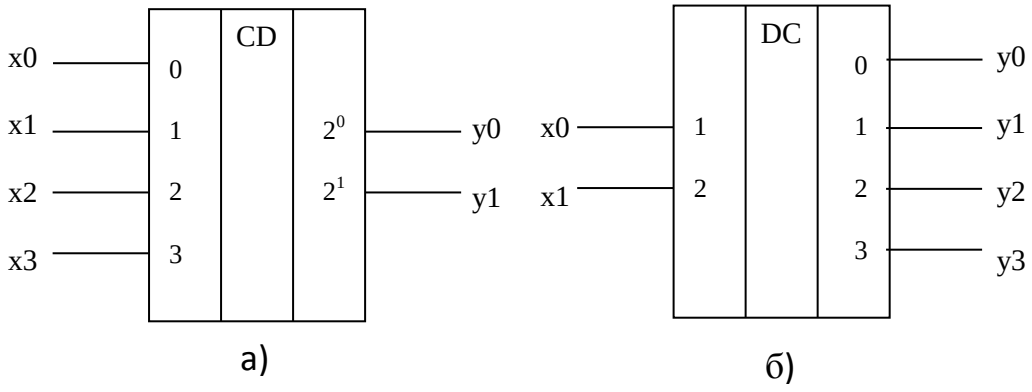


Рис. Б.1.Умовне графічне позначення; а) шифратора, б) дешифратора

В ЕОМ дешифратори широко використовуються для вибірки символів друкуючих пристроїв, комірок запам'ятовуючих пристроїв, розшифровки кодів операцій з видачею відповідних керуючих сигналів.

Б.2. МУЛЬТИПЛЕКСОРИ І ДЕМУЛЬТИПЛЕКСОРИ

Мультиплексори і демультиплексори – це комбінаційні пристрої, які належать до схем комутації.

У цифрових пристроях часто виникає завдання передачі цифрової інформації від *m* різних пристроїв через канал загального користування і зворотне завдання – передачі інформації із загального каналу *m* різним пристроям. Для цього на вході каналу встановлюється пристрій, який називається мультиплексором, а на виході каналу – пристрій, який називається демультиплексором.

Мультиплексор – пристрій, що здійснює підключення одного з вхідних каналів, який має адресу A_m , до загального вихідного каналу.

Демультиплексор – пристрій, який передає дані із загального вхідного каналу до одного з вихідних каналів, що має адресу *An*.

Роботу мультиплексора і демультиплексора можна спрощено представити за допомогою багатопозиційного ключа. Код адреси задає перемикачу певне положення для сполучення з виходом одного з інформаційних входів, або навпаки.

Входи мультиплексора і демультиплексора діляться на дві групи: інформаційні і адресні. Якщо мультиплексор має n адресних входів, то кількість інформаційних входів буде 2^n, а демультиплексор аналогічно буде мати 2^n виходів.

Таблицю істинності мультиплексора наведено у таб. Б.3. Умовне графічне позначення мультиплексора показано на рис. Б.2а. Таблицю істинності демультиплексора наведено у таб. Б.4. Умовне графічне позначення мультиплексора показано на рис. Б.2б.

Таблиця Б.3. Таблиця істинності мультиплексора

Інформаційні входи: xi	Адресні входи: a1 a0	Вихід: y
x3x2x1x0	0 0	x0
	0 1	x1
	1 0	x2
	1 1	x3

Таблиця Б.4. Таблиця істинності демультиплексора

Інформаційний вхід: x	Адресні входи: a1 a0	Виходи: y3,y2 y1 y0
x	0 0	0 0 0 x
	0 1	0 0 x 0
	1 0	0 x 0 0
	1 1	x 0 0 0

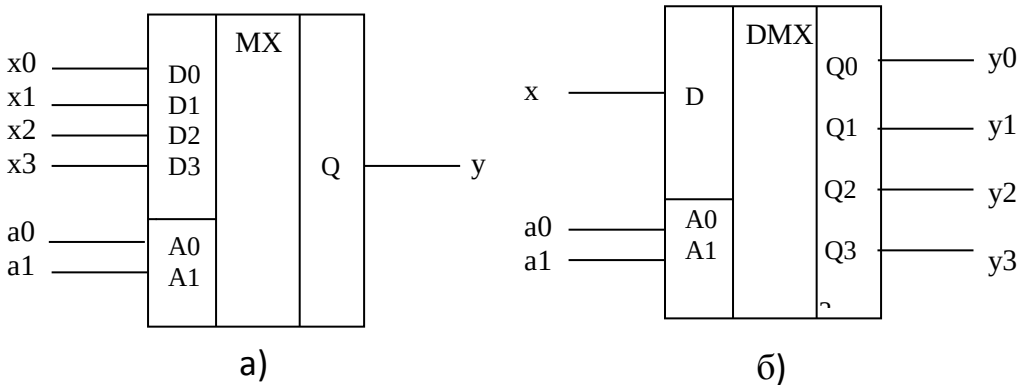


Рис. Б.2 Умовне графічне позначення; а) шифратора, б) дешифратора

ДОДАТОК В. ТРИГЕРИ

В.1. ЗАГАЛЬНІ ВІДОМОСТІ

Тригер — елемент статичної енергозалежної пам’яті, призначений для запису та зберігання 1 біта інформації (логічний 0, або 1), елементарний цифровий автомат Мура, який має 2 стани 0 та 1.

В цифрових пристроях на тригерах будуються послідовностні схеми (схеми, які мають внутрішню пап ять): регістри, лічильники та ін.

Тригери складаються з комірки пам'яті і схеми управління. Елемент пам'яті будується на двох інверторах, зв'язаних один з одним "навхрест" так, що вихід одного сполучений зі входом іншого. Сигнал «1» або «0», постійно інвертуючись, буде зберігатись в такому ланцюгу поки не буде вимкнуто живлення схеми.

Тригери мають два виходи: прямий Q і інвертований $\overline{Q}$. Стан тригера зчитується за значенням прямого виходу. Загалом тригери мають інформаційні входи, за допомогою яких записується біт даних, та входи управління записом (дозволяють чи забороняють запис з інформаційних входів в певний час).

За типом інформаційних входів розрізняють: RS, D, T, JK тригери. Використовуються також комбіновані тригери, в яких поєднуються одночасно декілька типів входів.

За способом запису інформації розрізняють асинхронні і синхронні тригери. Синхронні тригери мають управляючий вхід синхронізації С (від слова Clock). В асинхронних тригерах перехід в новий стан викликається безпосередньо змінами вхідних інформаційних сигналів. У синхронних тригерах перехід в новий стан відбувається тільки при подачі на вхід С дозволяючого синхросигналу.

За способом сприйняття тактових сигналів тригери діляться на статичні (керовані рівнем 0 або 1) і динамічні (керовані фронтом наростання або фронтом спаду) сигналу С.

Серед комбінованих тригерів найбільш поширені ті, у яких специфічні інформаційні входи, що синхронізується сигналом С, комбінуються з асинхронними R і S входами. В цьому випадку асинхронні входи установки і скидання є домінуючими – при їх дії сигнали на інших входах ігноруються. Асинхронні R і S входи зручні для швидкого встановлення тригерів в конкретний стан на початку або в процесі роботи пристрою. Як тільки ці сигнали отримують значення, відповідні режиму зберігання (R=S=0 або $\overline{R}$=$\overline{S}$=1), тригер продовжує працювати у синхронному режимі. Наприклад, тригер DRS з входом С при R=S=0 працює як D тригер з входом С, а при інших значеннях R і S – як асинхронний RS тригер. Таблиця істинності комбінованого D$\overline{R}$ $\overline{S}$ –тригера з синхронізацією за переднім фронтом С наведена у таб.В.1

Таблиця В.1. Таблиця істинності комбінованого D$\overline{R}$ $\overline{S}$ –тригера з синхронізацією за переднім фронтом С

$\overline{R}$	$\overline{S}$	C	D	Q	$\overline{Q}$
1	0	-	-	1	0
0	1	-	-	0	1
0	0	-	-	*	*
1	1	⌟⌜	0	0	1
1	1	⌟⌜	1	1	0
1	1	⌝⌞	-	Q’	$\overline{Q}$’
1	1	0	-	Q’	$\overline{Q}$’
1	1	1	-	Q’	$\overline{Q}$’

- будь-яке значення, * – заборонена комбінація сигналів

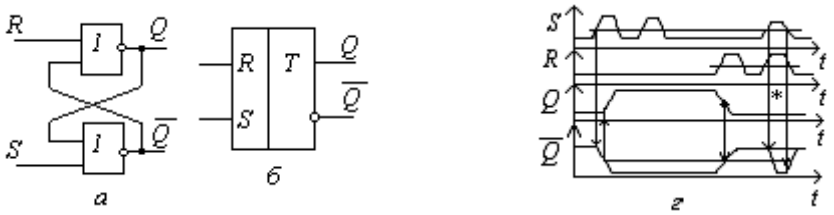
В.2. СХЕМОТЕХНІКА ТРИГЕРНИХ ПРИСТРОЇВ

В.2.1. RS- тригер

Асинхронний RS – тригер має 2 входи: R – вхід скидання тригера в 0 (Reset – скидання) і S – вхід встановлення тригера в 1 (Set – встановлення). На рис. В.1 приведена реалізація RS-тригера на елементах АБО—НІ, а на рис. В.2. – на елементах І-НІ. В таблиці істинності з апострофом Q’ позначено попереднє значення вихідного сигналу, а без апострофу Q – поточне значення.

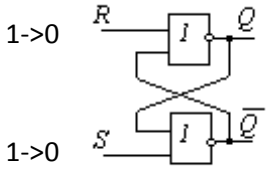
З часових діаграм видно, що тривалість перемикання $t_{пер}$ і мінімальна тривалість вхідного сигналу даних $t_{вх.min}$ для асинхронних RS-тригерів визначаються середнім часом затримки сигналу в логічних елементах

$$t_{зт.р.сер.}=0.5(t_{зт.р}^{01}+ t_{зт.р}^{10})$$
$$t_{пер}= t_{вх.min}=2t_{зт.р.сер}. \qquad \text{(В.1.)}$$



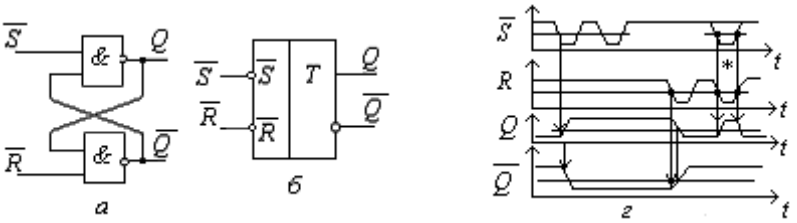
R	S	Q	$\overline{Q}$	Режим
1	0	0	1	Запис 0
0	1	1	0	Запис 1
0	0	Q’	$\overline{Q}$’	Зберігання
1	1	*	*	Заборонено

в



д

Рис. В.1. RS-тригер на елементах АБО—НІ: а) функціональна схема; б) умовне графічне зображення; в) таблиця істинності г) часові діаграми роботи; д) режим генерації



$\overline{R}$	$\overline{S}$	Q	$\overline{Q}$	Режим
0	1	0	1	Запис 0
1	0	1	0	Запис 1
1	1	Q’	$\overline{Q}$’	Зберігання
0	0	*	*	Заборонено

в

Рис. В.2. RS-тригер на елементах І-НІ : а) функціональна схема; б) умовне графічне зображення; в) часові діаграми роботи

0 1 0 Запись 1

1 1 Q Хранение

0 0 * * Запрещенное

На часових діаграмах символом (*) помічені інтервали дії на входи RS-тригерів заборонених комбінацій вхідних сигналів. При цьому в тригері на елементах АБО—НІ встановлюються вихідні рівні $Q = \overline{Q} = 0$, а в тригері на елементах І-НІ – $Q = \overline{Q} = 1$. Після закінчення дії забороненої комбінації і подачі на входи RS-тригера комбінації 01 або 10, тригер перейде в стан 0 або 1.

При строго одночасній зміні вхідних сигналів із забороненої комбінації в комбінацію, відповідну режиму зберігання, і при однакових затримках його елементів тригер переходить в режим генерації. Елементи Або-Ні чи І-НІ постійно будуть інвертувати свої значення, поки на входи не буде подана комбінація сигналів, яка встановлює тригер в 0 або 1.

На практиці завжди має місце випадковий розкид затримок логічних елементів, тому тригер все ж таки перейде в якийсь зі стійких станів («0» або «1»).

Асинхронний S-тригер, R-тригер і Е-тригер

Для усунення неоднозначності у роботі RS-тригера створено тригери, які при забороненій комбінації вхідних сигналів перемикаються: S-тригер – в одиничний стан, R-тригер – в нульовий стан і Е-тригер – зберігає попередній стан (від слова Exclusive).

Схема реалізації R-тригера в базисі І-НІ і часові діаграми його роботи приведені на рис. В.3. До схеми звичайного RS – тригера доданий зворотний зв'язок з виходу елементу DD2 на вхід логічного елементу DD1. Коли $S = R = 1$ вихід DD2, на якому встановлюється $\overline{R} = 0$, блокує решту входів елементу DD1 і встановлює на його виході логічну «1». Тому комбінація S=R=1 встановлює тригер в стан «0». Як видно з часових діаграм (рис. В.3б) додаткові логічні елементи DD1 і DD2 збільшують час перемикання R-тригера і мінімальну тривалість вхідного сигналу в порівнянні з виразом В.1:

$$t_{пер} = t_{вх.min} = 3t_{зт.р.сер}. \qquad (В.2)$$

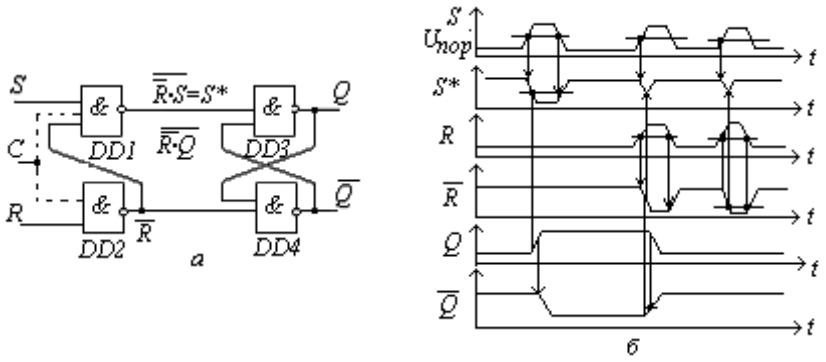


Рис. В.3. R-тригер на елементах І-НІ а) функціональна схема; б) часові діаграми роботи

На рис. В.4 показана схема S-тригера в базисі І-НІ. Зворотний зв'язок з виходу елементу DD1 на вхід елементу DD2 забезпечує пріоритет входу S, оскільки на вхід DD2 поступає рівень $\overline{S} = 0$, який для логічного елементу І-НІ є домінуючим і встановлює на виході елементу DD2 логічну «1».

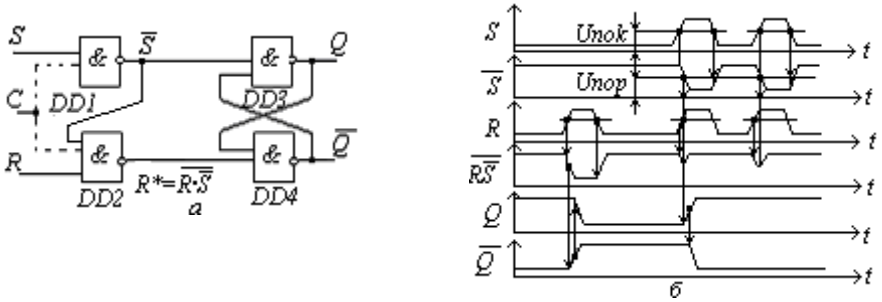


Рис. В.4. S-тригер на елементах І-НІ а) функціональна схема;б) часові діаграми роботи

Схема, що реалізовує Е-тригер в базисі логічних елементів І-НІ, і його часові діаграми показані на рис. В.5. Додаткові інвертори DD5 та DD6 при S=R=1 сигналами $\overline{S}=\overline{R}=0$ встановлюють на виходах елементів DD1, DD2 рівні логічної «1», що відповідає режиму зберігання раніше записаних даних.

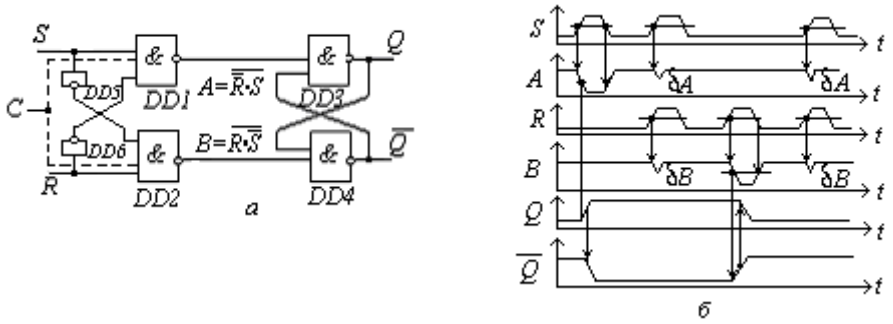


Рис. В.5. Е-тригер на елементах І-НІ а) функціональна схема; б) часові діаграми роботи

Унаслідок затримки вхідних сигналів інверторами DD5, DD6 на виходах вентилів DD1, DD2 формуються сигнали перешкоди δA, δB, аналогічно перешкодам в асинхронних тригерах типу R і S. Ці перешкоди за певних умов можуть привести до помилкового спрацьовування тригерів на елементах DD3 і DD4.

Синхронні R, S, E і RS-тригери, синхронізовані рівнем сигналу C

Для усунення помилкових спрацьовувань в R, S і E-тригерах необхідно синхронізувати їх роботу серією тактових імпульсів C. Для цього в схемах (рис. В.3, В.4, В.5) передбачається додатковий вхід C (показаний штриховою лінією). ***Рівні дозволу C повинні поступати з деякою затримкою t_{SU} щодо зміни інформаційних сигналів R і S, щоб обидва входи до їх запису встигали гарантовано перемкнутися.*** У E-тригері дозвіл C необхідно включати із додатковою затримкою на час перемикання DD5, DD6, а у R та S тригерах – на час перемикання одного з елементів DD1 або DD2. Таким чином, отримують варіанти синхронних R, S і E тригерів, які мають вищу надійність, але і більшу затримку перемикання.

Окрім боротьби з перешкодами режим синхронізації RS-тригерів широко використовується при побудові синхронних цифрових пристроїв. На рис. В.6 і В.7 показана реалізація синхронних RS-тригерів в базисах логічних елементів АБО-НІ та І-НІ. Схеми синхронізації побудовані на логічних елементах DD1, DD2 формують сигнали R*, S*, які керують

асинхронними RS-тригерами на логічних елементах DD3, DD4. Інформація, що поступає на входи S і R, як видно з часових діаграм, сприймається тільки на інтервалі дії дозволяючих синхроімпульсів C тривалістю t_c. Решту часу тригер знаходиться в режимі зберігання раніше записаної інформації.

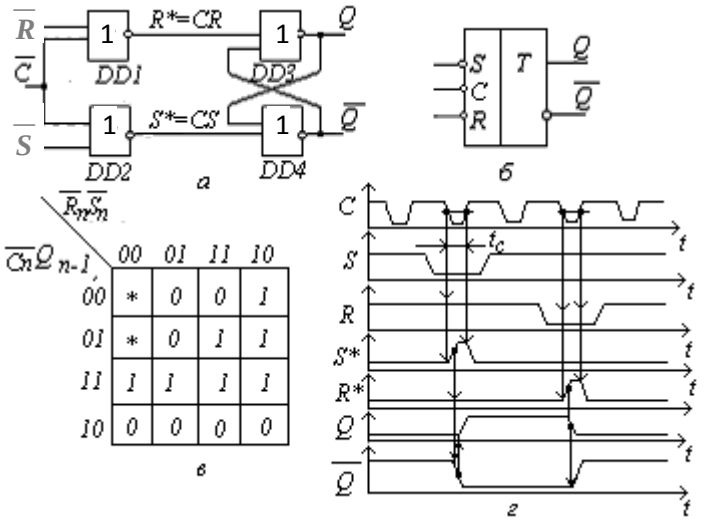


Рис. В.6 Синхронний RS-тригер на елементах АБО—НІ: а) функціональна схема; б) умовне графічне зображення; у) карта Карно; г) часові діаграми роботи

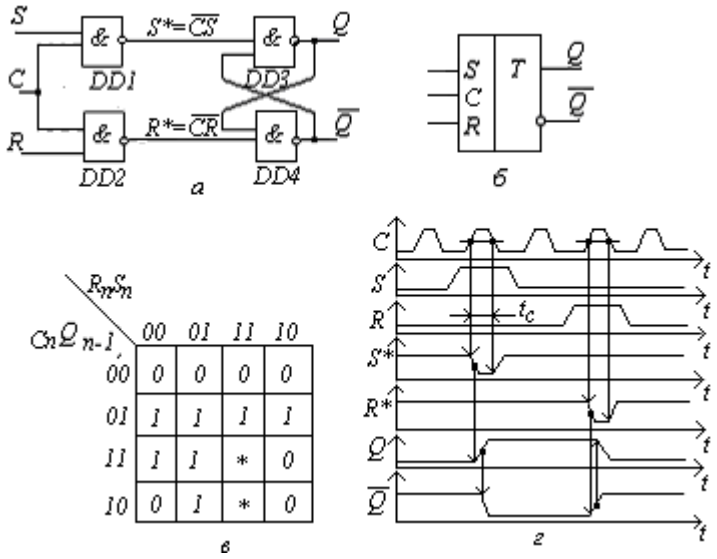


Рис. В.7. Синхронний RS-тригер на елементах І–НІ: а) функціональна схема; б) умовне графічне зображення; в) карта Карно; г) часові діаграми роботи

Всі зміни інформаційних сигналів R і S ($\overline{R}$ і $\overline{S}$) в інтервалі t_c передаються на вхід асинхронного тригера на елементах DD3, DD4 і викликають перемикання його станів (прозорий режим асинхронного запису), що характерний для всіх схем, що синхронізуються рівнем С. Аналогічно асинхронним тригерам, забороненою є комбінація вхідних сигналів R=S=1 ($\overline{R} = \overline{S} = 0$).

Тривалість перемикання $t_{пер}$ і мінімальна тривалість вхідного сигналу $t_{вх.min}$ для RS-тригеру, синхронізованому рівнем сигналу С складає:

$$t_{пер} = t_{вх.min} = 3t_{зт.\ р.\ сер,} \qquad (В.3)$$

де $t_{зт.\ р.\ сер}$, — середня затримка розповсюдження одного вентиля.

<u>Синхронні RS-тригери, синхронізовані фронтом сигналу С</u>

У динамічних RS-тригерах, що синхронізуються фронтом сигналу С, відсутній режим прозорого асинхронного запису. Інформаційні сигнали S і R можуть перемикатися багато разів, але тригер перемикається в стан, відповідний комбінації вхідних сигналів S і R безпосередньо перед дозволяючим (зростаючим або спадаючим) фронтом синхроімпульсу.

Для побудови динамічних тригерів будь-яких типів найчастіше використовуються 3 типи схемних рішень:

- ***2-х ступенева схема;***
- ***6-елементна схема;***
- ***схема "із забороннимизв'язками".***

Так звана ***6-елементна схема*** принципово включає елемент пам’яті, який протягом часу tc дії синхроімпульсу забезпечує управління станом асинхронного RS-тригера. У RS-тригерах (рис. В.8) схеми синхронізації тригерів побудовані на логічних елементах DD1 – DD4 типу АБО-НІ (а) або І-НІ (в). RS-тригер на елементах АБО-НІ синхронізується заднім (спадаючим) фронтом С, а на елементах І-НІ – переднім (зростаючим) фронтом С. Пунктирною лінією на схемі показаний принцип підключення входів асинхронної RS установки/скидання тригера, які домінують над інформаційними входами і входом синхронізації.

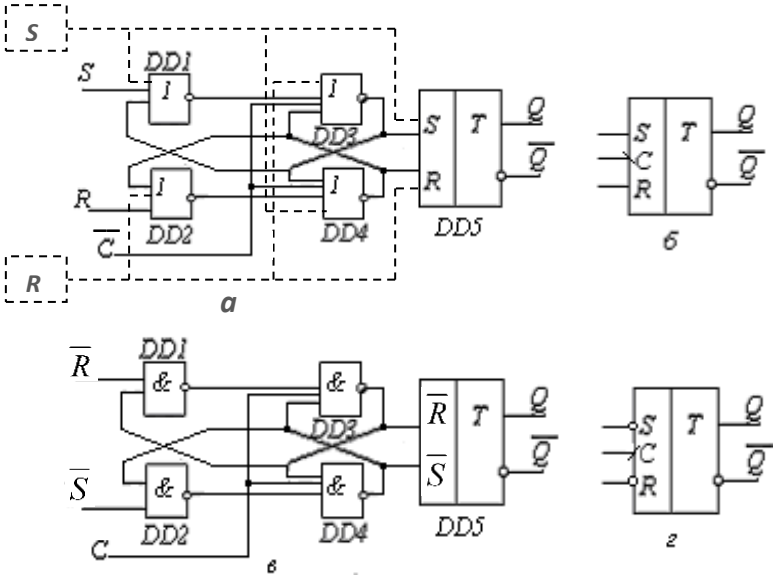


Рис. В.8. RS-тригер з динамічним управлінням (6-елементна схема): а) схема на елементах АБО—НІ ; б) умовне графічне позначення; в) схема на елементах І-НІ ; г) умовне графічне позначення

Розглянемо роботу схеми на елементах АБО—НІ (рис. В.8,а). Коли С=1 на виходах елементів DD3 і DD4 підтримуються рівні логічного «0», що забезпечує режим зберігання для асинхронного тригера DD5. Входи S і R не впливають на стан DD3, DD4, оскільки сигнал С = 1 є домінуючим.

Нехай безпосередньо перед заднім фронтом синхроімпульсу на інформаційних входи подано комбінацію вхідних сигналів S = 1, R =0. Це призведе до встановлення DD1=0 та DD2=1 через час перед встановлення тригера (t_{su}). Тоді при перемиканні входу С з «1» в «0» на виході DD4 зберігається «0», а на трьох входах елементу DD3 з'явиться «0», що приведе до встановлення на його виході «1». Цей сигнал «1» подається на входи DD1 та DD4 і блокує їх від перемикання зовнішніми сигналами. Вхідні сигнали повинні утримуватись незмінними до надійного блокування входів DD1, тобто більше затримки DD3 (t_h). Далі протягом часу tc дії рівня С = 0 тригер не буде реагувати на перемикання входів S і R. Оскільки на входах DD5 і DD6 встановлено S = 1, R =0, то тригер встановлюється в стан «1». Аналогічно відбувається встановлення тригера

в стан «0». Після наступного переключення С в «1», а DD3 – в «0» входи DD1 будуть знову розблоковані.

Комбінація сигналів S=R=1 є забороненою. Оскільки в цьому випадку після заднього фронту С тригер на елементах DD3, DD4 переходить в нестабільний режим (генерації), і це автоматично переноситься в тригер на DD5, DD6. При S=R=0 (режим зберігання), на виході DD1 встановиться стан інверсний DD3, а на виході DD2 – інверсний DD4. Тому стан DD3 і DD4 не міняється, отже і стан DD5 залишається попереднім.

Проте слід зазначити, що для правильної роботи схеми в режимі зберігання (S=R=0), зміну вхідних сигналів S і R можна проводити тільки при С=1, інакше можливий режим наскрізного управління, коли тригер від зміни сигналів S і R перемикатиметься асинхронно. Це відбувається тому, що перед заднім фронтом С DD3= DD4 =0, а після заднього фронту С при S=R=0 ці значення зберігаються. В такому випадку після заднього фронту С входи S і R не блокуються (на зворотних зв'язках немає жодного сигналу «1»), і всі зміни сигналів S і R при С=0 будуть вільно записуватись у DD3, DD4, а потім – в DD5.

Аналогічно працює і RS-тригер, що синхронізується переднім фронтом на елементах І-НІ (рис. В.8в).

Тривалість перемикання $t_{пер}$, мінімальна тривалість вхідного сигналу $t_{вх.min}$, час перед встановлення та витримки для 6-ти елементного RS-тригера складає:

$$t_{пер} = 3t_{зт\ р.\ сер}$$

$$t_{su} = 1t_{зт\ р.\ сер,}; \quad t_h > 1t_{зт\ р.\ сер,} = 2\ t_{зт\ р.\ сер;}$$

$$t_{вх\cdot min} = t_{su} + t_h = 3t_{зт\ р.\ сер,} \qquad \text{(В.4)}$$

де $t_{зт\ р.\ сер}$, — середня затримка розповсюдження одного вентиля.

Іншим варіантом динамічної реалізації RS-тригера є ***двоступенева схем Master-Slave.*** Прийом інформації у вхідний і вихідний ступені тригера виконується по черзі. Двоступенева схема має малий час витримки t_H, необхідний для блокування інформаційних входів після дозволяючого

фронту С. На її основі можна реалізувати різні типи тригерів без наскрізного управління і небезпечних змагань сигналів. Однак, вона складніша, ніж 6- елементна схеми, та має трохи нижчу швидкодію.

Двоступеневі тригери будуються декількома способами: з різнополярним управлінням ступенями (рис. В.9а), з інвертором (рис. В.9б). В обох варіантах ступені Master і Slave синхронізуються почергово. У другому варіанті для почергового управління ступенями у ланцюг тактових сигналів включений інвертор.

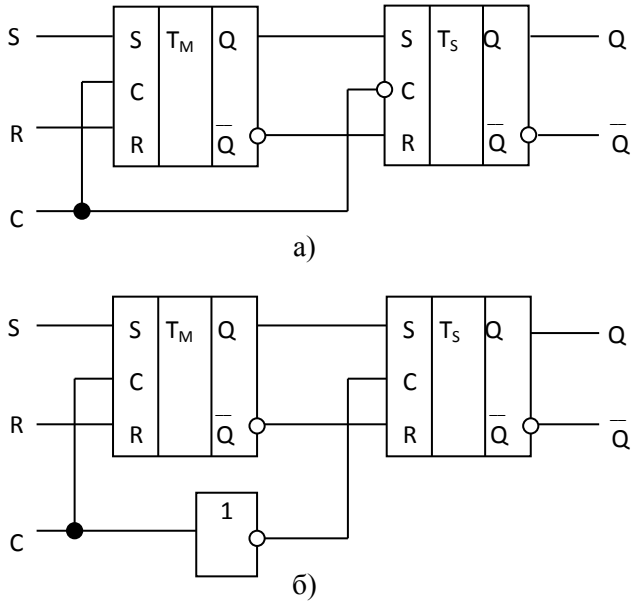


Рис. В.9. Схема двоступеневого тригера: а) з різнополярним управлінням; б) з інвертором

Організація *асинхронного RS встановлення* ***в комбінованому тригері*** на основі двоступеневої схеми (рис. В.9) аналогічна показаній на рис. В.15. Асинхронні RS-входи включається як треті входи в логічні елементи комірок пам'яті обох ступенів тригера.

Тривалість перемикання $t_{пер}$, мінімальна тривалість вхідного сигналу $t_{вх.min,}$ час перед встановлення, витримки та повний час розповсюдження вхідних сигналів до виходів для двоступеневого RS-тригера складає:

$t_{пер} = 4t_{зт\ р.\ сер}$;

$t_{su} = 3t_{зт\ р.\ сер}$; $t_h = 1t_{зт\ р.\ сер}$;

$t_{вх.min} = t_{su} + t_h = 4t_{зт\ р.\ сер}$;

$t_{роз} = 7t_{зт\ р.\ сер}$ (В.5)

де $t_{зт\ р.\ сер}$ — середня затримка розповсюдження одного вентиля.

У якості RS- тригера по *схемі "із забороннимиі зв'язками"* може бути використаний JK- тригер (відмінності в роботі тригерів спостерігатимуться тільки у разі комбінації вхідних сигналів R=S=1, яка для RS- тригера є забороненою). Приклад схеми на елементах І-НІ приведений на рис В.15. У чистому виді (без зворотних зв'язків з виходів М-ступені на входи S-ступені) RS тригер по схемі «із забороннимиі зв'язками» не будується, оскільки у разі комбінації сигналів R=S=0, можливий прозорий режим асинхронного запису.

В.2.2. D- тригер

D тригер (від слова Delay – затримка) має один інформаційний вхід D і вхід синхронізації С. В моменти дії дозволяючого рівня або фронту синхросигналу тригер встановлюється в стан, відповідний логічному рівню сигналу на вході D, а після завершення дії дозволяючого синхросигналу переходить в режим зберігання інформації. Таким чином, тригер типу D реалізує затримку вхідного сигналу. Практичне застосування знаходять лише синхронні D-тригери.

D-тригери, які синхронізуються рівнем синхросигналу

Реалізації синхронних D тригерів, які синхронізуються рівнем синхросигналу, на елементах АБО-НІ та І-НІ показані на рис. В.10.

Якщо на синхронізуючий вхід D тригера (рис. В.10,а) поданий рівень $\overline{C}$=1 (або С=0 для рис. В.10,г), який є домінуючим для логічних елементів DD1, DD2, то на їх виходах встановлюються рівні R* =S*=0 ($\overline{R}$*= $\overline{S}$*=1). Ці рівні забезпечують режим зберігання для асинхронного тригера на елементах DD3, DD4 незалежно від стану інформаційного входу D. При $\overline{C}$ =0 (С=1) інформаційний вхід однозначно визначає стан виходу елементу

DD1, який у свою чергу обумовлює інверсний рівень на виході елементу DD2, і в тригер записується сигнал, поданий на вхід D.

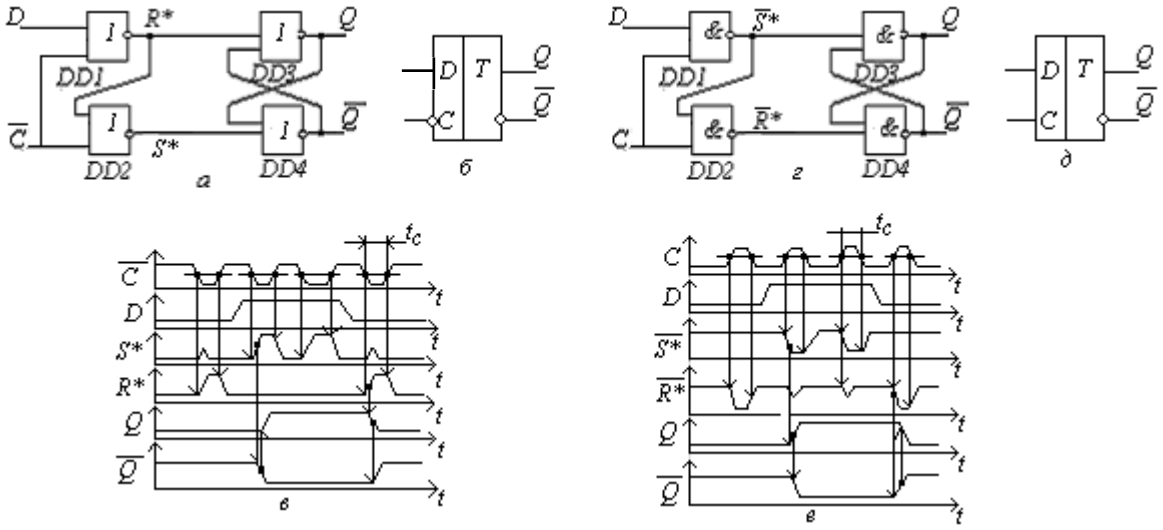


Рис. В.10. Синхронні D-тригери: а) функціональна схема на елементах АБО—НІ; б) умовне графічне зображення; в) часові діаграми роботи; г) функціональна схема на елементах І-НІ ; д) умовне графічне зображення; е) часові діаграми роботи.

Для D тригерів, які синхронізуються рівнем сигналу С, перемикання стану відбувається при перемиканні інформаційного входу D на всьому інтервалі дозволу С. Для того, щоб тригер фіксував значення сигналу з D входу тільки в певний момент часу використовується синхронізація фронтом синхроімпульсу.

D - тригер, що синхронізується фронтом синхросигналу

D тригер на елементах І-НІ, що синхронізується фронтом С, показаний на рис. В.11. Він побудований ***по 6-елементній схемі*** (на трьох асинхронних тригерах), з них 2 тригери на елементах DD1– DD4 утворюють схему синхронізації для основної комірки пам'яті на елементах DD5, DD6. Аналогічний тригер, можна реалізувати на елементах АБО-НІ.

З часових діаграм роботи D тригера (рис. В.11 в) видно, що при С=0 на виходах А і В підтримуються рівні «1» незалежно від стану входу D. Це відповідає режиму зберігання інформації в основному тригері на

елементах DD5, DD6. Перемикання рівня на вході D впливає тільки на логічні стани виходів Е і Ж тригерів схеми синхронізації. Якщо D=0, то Ж=1, Е=0, а у разі D=1 встановлюється Ж=0, Е=1. При цьому один з тригерів схеми синхронізації знаходиться в стійкому стані, а інший – в режимі розриву тригерних зв'язків при рівнях логічної «1» на обох виходах. Наприклад, якщо D=0, то на виходах тригера на елементах DD1.3, DD1.4 встановлюються однакові рівні Ж=В=1.

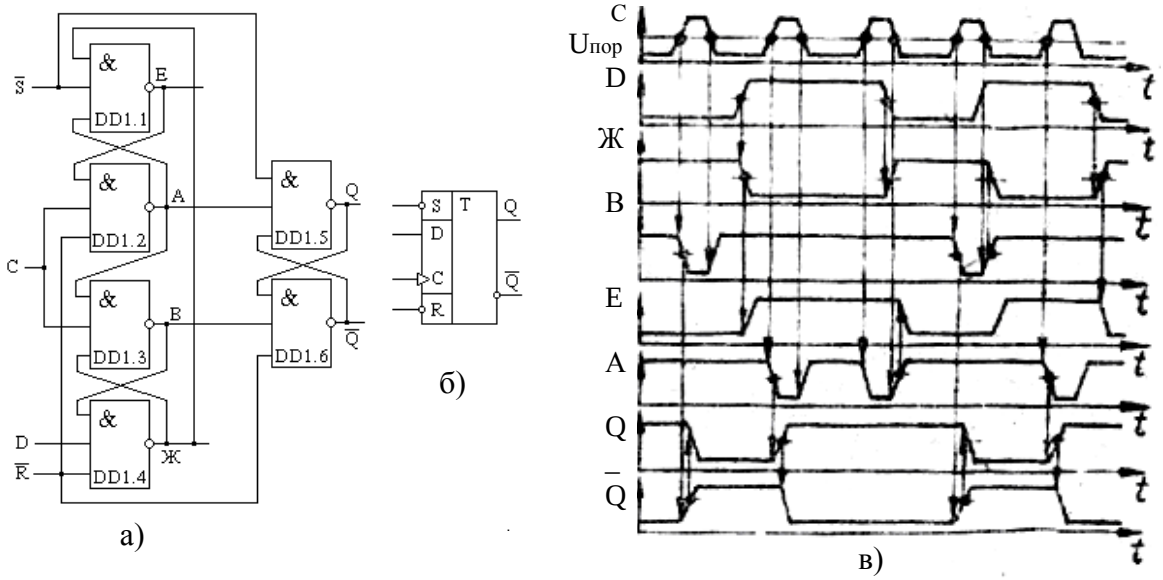


Рис. В.11. D тригер з динамічним управлінням (6-елементна схема): а) схема на елементах І-НІ; б) умовне графічне позначення; в) часові діаграми роботи

По передньому фронту синхроімпульсу С=0→1 тригер, що знаходився до цього в режимі розриву тригерних зв'язків, переходить в нормальний стійкий стан, і на входах основного тригера формуються взаємно інверсні логічні рівні. Якщо D=0, то В=0, А=1, і тригер встановлюється в стан «0»; якщо D=1, то В=1, А=0, і відбувається встановлення в стан «1».

З діаграм (рис.В.11,в) так само видно, що при С=1 перемикання стану інформаційного входу D не впливає на стан даного D- тригера. Це пояснюється тим, що при встановленні основного тригера в стан «0»

сигналом В=0 одночасно блокується логічний елемент DD1.4, і на його виході встановлюється Ж=1 незалежно від стану входу D. При встановленні основного тригера в стан «1» сигналом А=0 блокуються логічні елементи DD1.1 і DD1.3. Після цього перемикання сигналів D і Ж не впливає на стан основного тригера.

На рис. В.11,а показано ***комбінований D тригер*** з асинхронним встановленням за допомогою входів $\overline{R}$ та $\overline{S}$.

Часові параметри тригера аналогічні розглянутим раніше для 6-ти елементної схеми RS тригера (В.4) окрім:

$$t_{su} = 2t_{зт\ р.\ сер,\ ;} \qquad \text{(В.6)}$$

$$t_{вх.min} = t_{su} + t_h = 4t_{зт\ р.\ сер,}$$

де $t_{зт\ р.\ сер}$, — середня затримка розповсюдження одного вентиля.

Ще один спосіб побудови динамічного D тригера полягає у використанні ***двоступеневих схем MS-типа*** (рис.В.12).

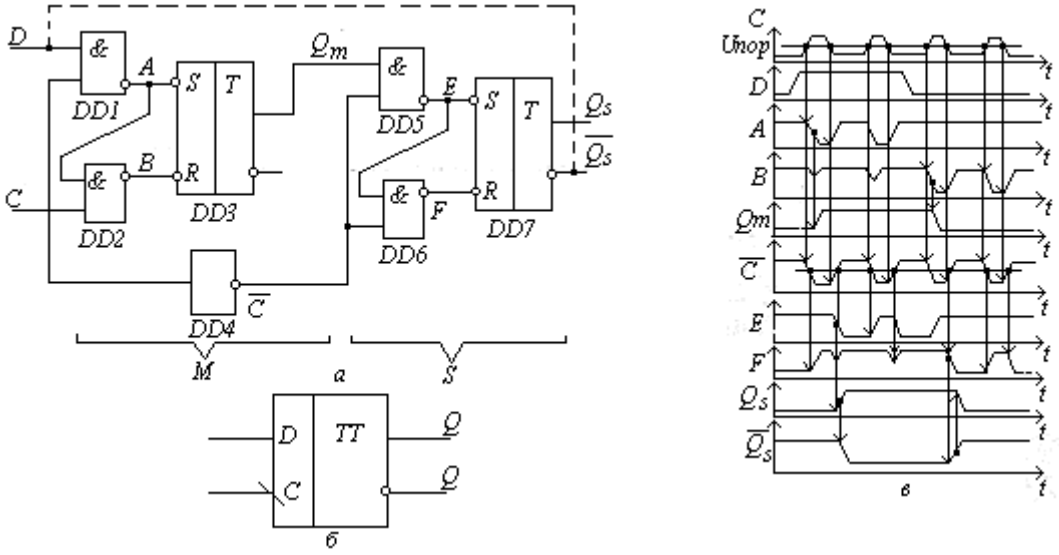


Рис.В.12. D-тригер з динамічним управлінням (двоступенева схема): а) схема на елементах І-НІ; б) умовне графічне позначення; в) часові діаграми роботи

Завдяки синхронізації протифазними імпульсами C і $\overline{C}$ запис нової інформації в тригери M і S ступенів принципово розділений в часі, що виключає наскрізну передачу інформації з входу D на виходи Qs, $\overline{Q}$s.

При C=0 тригер допоміжного ступеня M знаходиться в режимі зберігання даних, а у вихідний тригер рівнем $\overline{C}$ =1 дозволяється перезапис даних з тригера M. Переключення синхроімпульсу C=0→1 і $\overline{C}$=1→0 змінює режим роботи тригерів M і S. Тепер тригер M переходить в режим запису інформації з входу D, а тригер S – в режим зберігання інформації, записаної на попередньому кроці. У допоміжному тригері M можливий режим асинхронного багатократного запису на інтервалі дії синхроімпульсу C=1. При перемиканні синхроімпульсу C=1→0 інформаційний вхід D блокується, і в основний тригер S переписується останній стан Q_M.

Часові параметри тригера ідентичні розглянутим раніше для двоступеневого RS-тригера (В.5).

D–тригер може бути реалізований і по *схемі* ***"із забороннимизв'язками"***з додатковим інвертором, як показано на рис. В.15б. При цьому зворотні зв'язки в схемі рис. В.15а з виходів Q і $\overline{Q}$ на входи можна видалити, оскільки на відміну від RS тригера в D–тригері на входи елементів DD1 і DD2 завжди подаватимуться взаємно інверсні значення сигналів і комбінація сигналів R=S=0, яка може викликати прозорий режим асинхронного запису, буде відсутній (так само як і заборонена комбінація R=S=1).

На рис. В.15а також показана організація асинхронного RS встановлення в ***комбінованому тригері***.

В.2.3. Т- тригер

Т–тригери мають один вхід Т, і з кожним вхідним сигналом перемикаються у протилежний стан.

Т тригери, керовані рівнем синхросигналу, зазвичай не реалізуються, оскільки в цьому випадку при утриманні дозволяючого рівня

сигналу на Т – вході тригер багато разів перемикатиметься в протилежний стан, тобто буде знаходитись у режимі генерації.

Т тригери, керовані фронтом синхросигналу

Таблиці істинності синхронного (за переднім фронтом) Т тригера та Т тригера з динамічним Т входом (керованим переднім фронтом сигналу Т) показано в таб. В.2.

Таблиця В.2. Таблиці істинності Т тригера: а) синхронного; б) з динамічним Т входом

а)

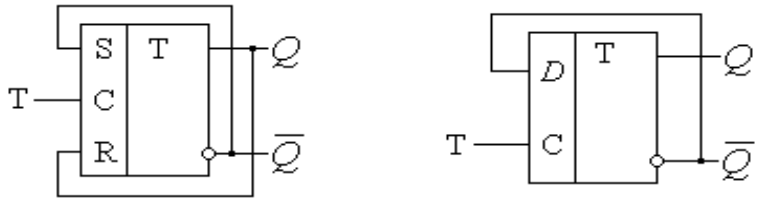
C	**T**	**Q**
⌐¬	1	$\overline{Q}$,
⌐¬ ¬⌙ 0, 1	0 - -	Q’

б)

T	**Q**	**Режим**
⌐¬ (0→1)	$\overline{Q}$,	Інверсія попереднього
0	Q’	Зберігання
1		
¬⌙ (1→0)		

Т тригери можуть бути побудовані на основі трьох, розглянутих раніше, базових схем. На рис В.13. показаний принцип побудови Т тригера на основі RS і D тригерів з динамічною синхронізацією для виключення режиму генерації.

Оскільки Т тригер має зворотні зв'язки зі своїх виходів на входи, і може знаходитись або в режимі зберігання, або в режимі інверсії попереднього значення, то його зазвичай необхідно встановлювати у певний початковий стан за допомогою додаткових входів асинхронного RS встановлення.



а) б)

Рис. В.13. Т тригер: а) схема побудови на основі RS тригера; б) схема побудови на основі D тригера

Схема побудови Т-тригера ***двоступінчатого MS-типа*** на основі відповідного D-тригера приведена на рис В.12,а (показано штриховою лінією). Для цього необхідно інформаційний вхід D-тригера на рис. В.12,а з'єднати з інверсним виходом $\overline{Q}$ як на рис В.13,б, а на вхід синхронізації С подавати імпульси Т. Аналогічно можна побудувати схеми Т-тригерів на основі схем, зображених на рис. В.9, якщо з'єднати їх R і S входи з виходами Q і $\overline{Q}$ відповідно до рис.В.13,б.

Схема ***«6-ти- елементного» типу»*** будується за принципом, показаним на рис.В.13. на основі схем рис.В.8 (RS тригер) або рис.В.11 (D тригер).

Т тригер може бути побудований по схемі ***«із забороннимизв'язками»*** (рис. В.14,а).

Сигнали блокування другого ступеня беруться в цій схемі з входів фіксатора першого ступеня. Перемикання тригерів ступенів М і S розділені в часі самим сигналом Т. Тригер ступеня М перемикається одиничним, а тригер ступеня S – нульовим рівнем сигналу Т. Тобто, стан MS тригера змінюється по кожному задньому фронту сигналу Т.

Реалізація розділеного в часі режиму роботи тригерів ступенів М і S досягається використанням зв'язків виходів елементів DD1 (А) і DD2 (В) з входами елементів DD4, DD5. Запис інформації в тригер ступеня М одночасно блокує запис в тригер ступеня S. Оскільки в цьому випадку на входи елементів DD1 і DD2 завжди подаються взаємно інверсні значення сигналів з виходів тригера ступеня М, то при Т=1 на одному з виходів А або В обов'язково встановлюється логічний «0», який блокує інші входи

елементів DD4, DD5. На їх виходах E і F встановлюються значення логічної «1», що відповідає режиму зберігання для тригера ступеня S. І навпаки, якщо T=0, то на виходах елементів DD1 і DD2 встановлюються значення A=B=1. В цьому випадку тригер ступеня M перемикається в режим зберігання інформації, а вентилі DD4, DD5 розблоковуються для перезапису інформації з виходів тригера ступеня M (DD3) в тригер ступеня S (DD6).

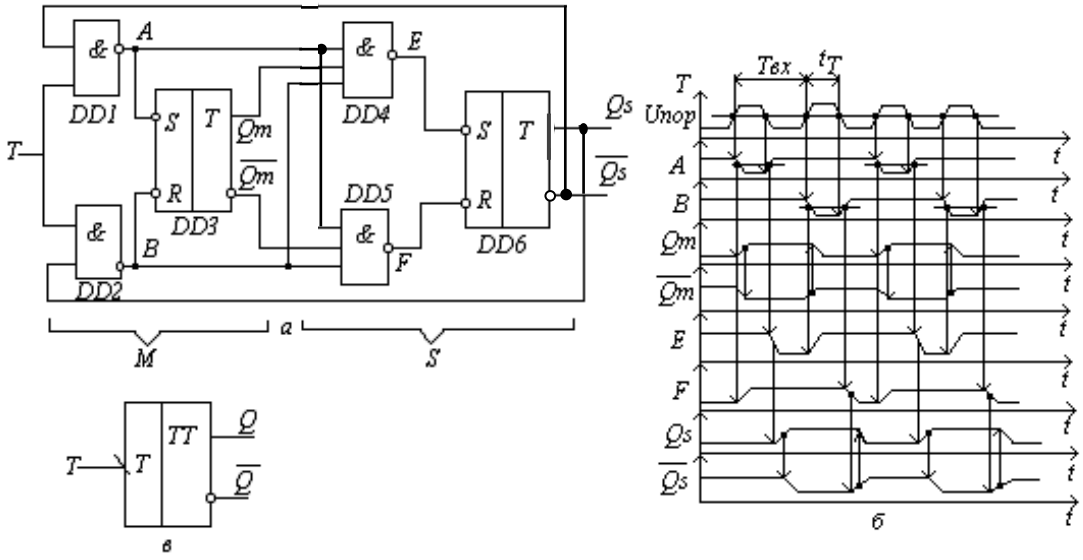


Рис. В.14. Т-тригер з динамічним управлінням: а) функціональна схема з «заборонними зв'язками» на елементах І-НІ; б) умовне графічне зображення; в) часові діаграми роботи.

Т тригер по схемі «із заборонними зв'язками» також можна легко побудувати на основі JK-тригер рис. В.15б.

Комбінований Т- тригер з асинхронними входами RS встановлення на основі схеми «із заборонними зв'язками» будується так само, як показано на рис. В.15а, прямим включенням додаткових входів $\overline{R}$, $\overline{S}$ в елементи І-НІ комірок DD3 та DD6.

Тривалість перемикання $t_{пер}$, мінімальна тривалість вхідного сигналу $t_{вх.min}$, час передвстановлення, витримки та повний час розповсюдження вхідних сигналів до виходів для схеми «із заборонними зв'язками» складає:

$$t_{пер} = 3t_{зт\ р.\ сер};$$
$$t_{su} = 3t_{зт\ р.\ сер};\quad t_h = 1t_{зт\ р.\ сер};$$
$$t_{вх.min} = t_{su} + t_h = 4t_{зт\ р.\ сер};$$
$$t_{роз} = 6t_{зт\ р.\ сер} \qquad \text{(В.7)}$$

де $t_{зт\ р.\ сер}$ — середня затримка розповсюдження одного вентиля.

Час перемикання і розповсюдження сигналу тут менше, ніж для двоступеневої MS схеми, на одну затримку інвертора в ланцюзі синхронізації.

Як видно з часових діаграм (рис В.14 б), частота імпульсів на виході Т тригера fвих = fвх/2. Таким чином, їх можна використовувати як дільників частоти на 2. При цьому максимальна частота вхідних імпульсів при мінімально допустимій їх тривалості складають:

$$f_{Tmax} = 1/(6\ t_{зт.р.сер});$$
$$t_T \geq 3\ t_{зт\ р.сер} \qquad \text{(В.8)}$$

де t_T – довжина логічної «1» на вході Т.

В.1.4. Універсальний JK-тригер

JK тригер – має два інформаційні входи: установки J (від jump) і скидання K (від kill). У разі вхідної комбінації J=K=1 тригер перемикається у протилежний стан подібно до Т тригера, а при будь-яких інших комбінаціях – працює як RS-тригер, у якого входу S відповідає вхід J, а входу R – вхід К. З тих же причин, що і Т тригер JK тригер зазвичай будується за двоступеневою схемою з синхронізацією по фронту С, і часто має входи початкового SR встановлення. JK тригер називається універсальним тому, що на його основі можна побудувати тригери інших типів (рис. В.15 б).

Таблиця істинності JK- тригера показано в таб.В.3.

*Таблиця В.3.*Таблиця істинності JK- тригера

K(=R)	**J(=S)**	**Q**	**$\overline{Q}$**	**Режим**
1	0	0	1	Запис 0
0	1	1	0	Запис 1
0	0	Q’	$\overline{Q}$ ’	Зберігання
1	1	$\overline{Q}$ ’	Q’	Т- режим

Функціонування JK-тригера ***«із забороннимиі зв'язками»*** (рис. В.15,а) повністю аналогічно роботі Т-тригера (рис.В.14).

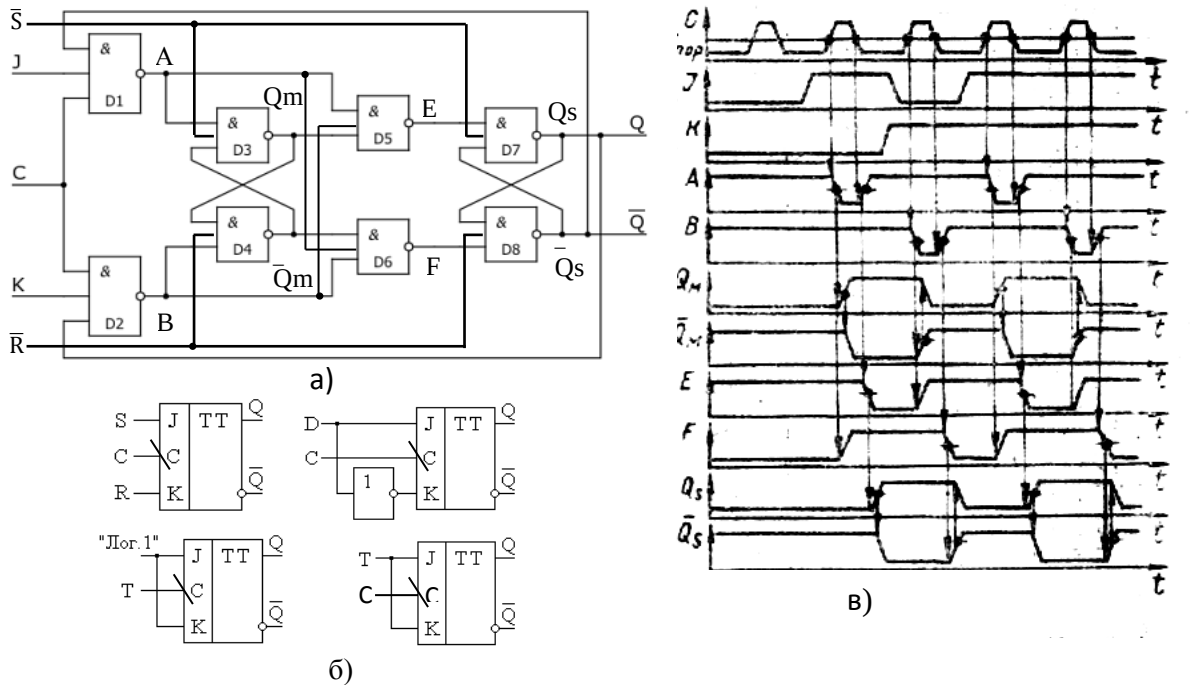


Рис. В.15. JK-тригер з динамічним управлінням (схема з «забороннимиі зв'язками» а) функціональна схема на елементах І-НЕ; б) умовне графічне зображення і способи побудови на його основі тригерів інших типів; в) часові діаграми роботи.

Діаграма на рис.В.14 відповідає умові: J=K=1, а C грає роль входу T. Тобто тригер перемикається у протилежний стан по задньому фронту сигналу C, що обумовлюється зворотними зв'язками з виходів Q і $\overline{Q}$ на входи тригера. У разі, коли J ≠ K можливі дві ситуації.

Перша ситуація: однакові сигнали на входах кожного з елементів D1(J=$\overline{Q}$), D2(K=Q), тоді входи J і K не впливають на роботу тригера, і він перемикатиметься у протилежний стан.

Друга ситуація: J≠$\overline{Q}$, K≠Q, що означатиме спробу перемкнути тригер в той стан, в якому він вже знаходиться. В цьому випадку сигнали A і B набувають значення логічної «1», оскільки один з вхідних сигналів елементу D1 (J або $\overline{Q}$) і елементу D2 (K або Q) буде рівний «0». На виходах цих елементів встановлюється A=B=1, що означає режим зберігання для комірки пам’яті на елементах D3, D4, а елементи D5 і D6 хоч і відкриваються для запису стану Qm у верхній ступінь Qs, але ці стани і так вже співпадають. Тому JK тригер залишиться у попередньому стані. У разі, коли J=K=0 (режим зберігання), встановлюється A=B=1, і стан тригера не змінюється.

Проте слід зазначити, що для правильної роботи схеми в цьому режимі зміну вхідних сигналів J і K можна проводити тільки при C=0. Інакше, якщо комбінація J=K=0 слідуватиме за J=K=1, то замість режиму зберігання матимемо перемикання в протилежний стан. Ця проблема наголошувалась раніше для RS-тригера, його не можна будувати по схемі з «забороннними зв'язками». Там вона виникає і при інших комбінаціях вхідних сигналів перед R=S=0. Це відбувається тому, що при J=K=0 тригер верхнього ступеня відкривається для переписування даних з нижнього ступеня, і в нього при C=1 встигають записатися нові дані, тобто Qm ≠ Qs.

Часові параметри JK тригера з «забороннними зв'язками» відповідають виразам В.7 та В.8.

JK – тригер по ***двоступеневій MS та по 6-ти елементній схемі*** може бути побудований на основі відповідних схем SR тригерів (рис.В.9 і В.8) з додаванням зворотних зв'язків по схемі на рис. В.16.

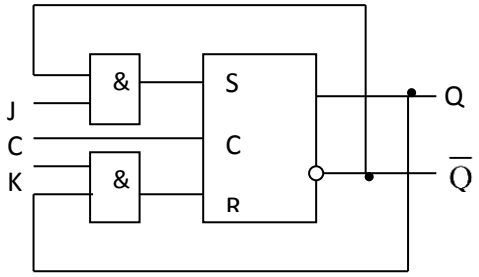


Рис. В.16. Схема побудови JK- тригера на основі SR- тригера

Останнім часом набули поширення і ***одноступеневі JK і T тригери з внутрішньою затримкою*** (рис. В.17, а). Функціонування одноступеневого тригера можна розглянути за допомогою часових діаграм (рис. В.17 б). На діаграмі показано процес переключення тригера з «0» в «1» при J=K=1. В цьому випадку J і K входи не впливають на роботу елементів 1 і 2, тому вони показані штриховими лініями. Оскільки схема симетрична, зворотне перемикання відбуватиметься аналогічно.

Підключення входів асинхронного встановлення/скидання $\overline{S}$ і $\overline{R}$ показано на рис. В.17а штриховими лініями. Коли тригер працює як синхронний JK тригер (при $\overline{S}=\overline{R}=1$) ці входи не впливають на роботу схеми.

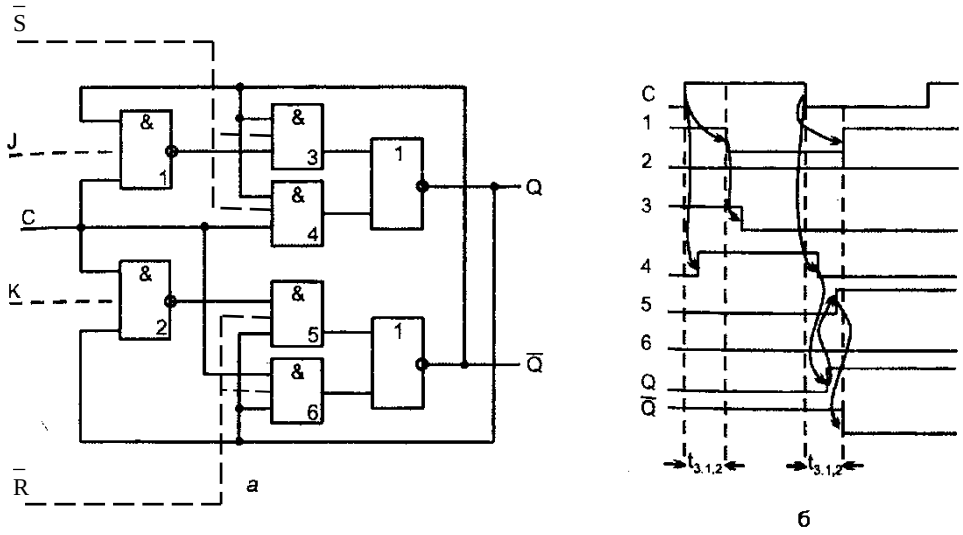


Рис.В.17. JK- тригер: а)схема з внутрішніми затримками; б) часова діаграма роботи схеми

Працездатність тригера забезпечується тільки за умови $t_{з1,2} > t_{з3,4,5,6} + t_{3або-ні}$ (затримки вентилів 1 і 2 перевищують сумарну затримку вентилів І + АБО-НІ), що відображено на часових діаграмах. Як видно з діаграми, тригер перемикається по задньому фронту синхросигналу.

Комірка пам'яті тригера будується на 2-х елементах АБО-НІ, які в сукупності з елементами 3 – 6 утворюють асинхронний $\overline{S}\,\overline{R}$ – тригер. Один інвертор – АБО-НІ + (І3 або І4) реалізує функцію Q = not (І1$\overline{Q}$' v С$\overline{Q}$'), другий інвертор – АБО-НІ +(І5 або І6) реалізує функцію $\overline{Q}$=not(І2Q'vCQ'). При С=0 на виходах І4 і І6 забезпечується логічний «0», і вони не впливають на комірку пам'яті, а на виходах І1 і І2 встановлюється логічна «1», що забезпечує режим зберігання для $\overline{S}\,\overline{R}$ – тригера на елементах АБО-НІ + І3, АБО-НІ + І5. При С=1 на виходах елементів І1 і І2 з'являється вхідна інформація, яка переписується в І3 або І5, але завдяки С=1 встановлюється Q=not (І1$\overline{Q}$'v$\overline{Q}$') =Q', $\overline{Q}$=not (І2Q'vQ') =$\overline{Q}$'. Таким чином, для $\overline{S}\,\overline{R}$ – тригера на елементах АБО-НІ + І4, АБО-НІ + І6 забезпечується режим зберігання. При подальшому перемиканні входу С в «0» елементи І4 і І6 знову перестають впливати на елемент пам'яті, і тепер Q може перемкнутися від І3 – Q =not (І1$\overline{Q}$'), а потім перемкнути І5 та $\overline{Q}$=not (І2Q). Аналогічно $\overline{Q}$ може перемкнутися від І5, і потім перемкнути І3 та Q. ише після того, як тригер встигне перемкнутися у новий стан, елементи І1 і І2 (вони мають велику затримку) переведуть його у режим зберігання.

ДОДАТОК Г.
ВИЗНАЧЕННЯ ДИНАМІЧНИХ ПАРАМЕТРІВ ТРИГЕРНИХ СХЕМ

Елементи, з яких складаються цифрові пристрої (ЦП), є інерційними (переключаються з затримками), що накладає обмеження на частоту і послідовність подачі сигналів на входи ЦП. Переважна більшість складних ЦП має внутрішню пам'ять. Таки пристрої відносяться до класу послідовностних схем. Функціональна стійкість (правильна робота) послідовностних схем ***гарантується тільки при дотриманні всіх припустимих значень їх часових параметрів.*** А враховуючи розкид затримок у реальних елементах, параметри вхідних сигналів у проектованих ЦП мають дещо перевищувати їх мінімально припустимі значення. ***тригерних схем***

В якості елементів пам'яті в ЦП широко використовують синхронні тригери, які є елементарними послідовностними пристроями. Тому для визначення часових параметрів ЦП, побудованих на тригерах, потрібно, в першу чергу, розуміти часові параметри тригерів.

Обмеження на частоту і послідовність подачі інформаційних та управляючих сигналів забезпечують відсутність «гонок сигналів» на входах тригерів, які можуть призводити до ризиків збою вихідних сигналів тригерів. Крім цього деякі схемні рішення тригерів накладають обмеження на певні комбінації вхідних сигналів. Наприклад, для переважної більшості RS- тригерів послідовність «заборонена комбінація RS» → «режим зберігання» призводить до нестабільної роботи тригера (режим генерації). Також до збоїв у роботі багатьох схем призводить перемикання входів асинхронного RS- встановлення одночасно с дозволяючим фронтом синхроімпульсу. Щодо останнього, то загальний принцип запобігання збоям в асинхронних цифрових автоматах (саме до них відносяться всі тригери) є ***правило – не змінювати декілька будь яких вхідних сигналів одночасно.***

Для визначення динамічних параметрів синхронних тригерів потрібно розуміти приципи їх роботи.

Г.1. ЗАГАЛЬНІ ПРИНЦИПИ РОБОТИ СИНХРОННИХ ТРИГЕРІВ

За характером процесу перемикання тригери поділяються на одноступеневі та двохступеневі. Тригери, керовані рівнем синхросигналу, завжди будуються за одноступеневою схемою, а тригери, керовані фронтом синхросигналу, як правило, будуються за двоступеневою схемою.

Синхронізація за рівнем синхросигналу в одноступеневих тригерах організована наступним чином. Коли синхросигнал на вході тригера приймає дозволяюче значення, тригер перемикається у новий стан при кожній зміні інформаційних сигналів. При забороняючому значенні синхросигналу на вході тригеру, він знаходиться у стані зберігання останніх записаних даних.

Двохступеневі тригери складаються з двох одноступеневих тригерів: вхідного ступеня – Master та вихідного – Slave. Перемикання тригера у новий стан відбувається в два етапи. Спочатку, один з рівнів тактового сигналу дозволяє прийом нових даних у вхідний ступень Master, а вихідний ступень Slave в цей час утримується в режимі зберігання попередніх даних. Потім, інший рівень тактового сигналу переводить Master у стан зберігання останніх вхідних даних, які були записані під час дозволу прийому, а Slave переводить у режим переписування даних з Master на вихід тригера.

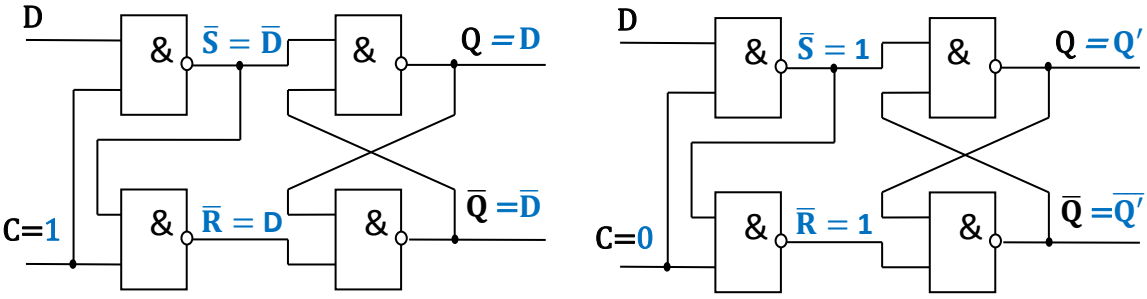
При аналізі роботи схем пам'ятаємо принцип домінуючого сигналу: A & 0 = 0, A & 1 = A; та Av1 = 1, Av0 = A.

Г.1.1. Принцип роботи тригерів, керованих рівнем синхросигналу

Принципова схема D-тригера на елементах 2І-НІ приведена на рис.Г.1. Тригер синхронізується рівнем С=1 і складається зі схеми

керування і комірки пам’яті на асинхронному $\overline{R}$ $\overline{S}$- тригері. Принцип запису значення сигналу D в тригер показано на рис. Г.1.

Сигнал C=1 не впливає на виходи елементів схеми керування, які повністю визначаються сигналом D: D&1=D, і т.д. Після встановлення на виходах елементів керування вхідний сигнал записується в комірку пам’яті ($\overline{S}$=$\overline{D}$, $\overline{R}$=D).



а) Режим запису

б) Режим зберігання

Рис. Г.1. Принцип роботи одноступеневого тригера: а) C=1 – дозвіл запису, б) C=0 – заборона запису

Сигнал C=0 блокує вхідний сигнал D, встановлюючи на виходах елементів схеми керування $\overline{R} = \overline{S} = 1$, що забезпечує режим зберігання в комірці пам’яті. ***Нехай всі логічні вентилі І-НІ мають однакову середню затримку tзр.*** Розглянемо два варіанти запису нового значення інформаційного сигналу в тригер.

На рис. Г.2 номерами показаний порядок поширення сигналів ***від зміни сигналу на вході D*** при наявності дозволу на вході C (C=1) ***до зміни сигналу*** на останньому з виходів (Q або $\overline{Q}$).

Прослідкуємо послідовність переключень: C=1 (дозвіл запису), D переключається з 0 в 1. Елементи переключаються в такому порядку: $\overline{S}$, потім одночасно $\overline{R}$ і Q, потім $\overline{Q}$. Таким чином, затримка тригера Tзтр=3 tзр.

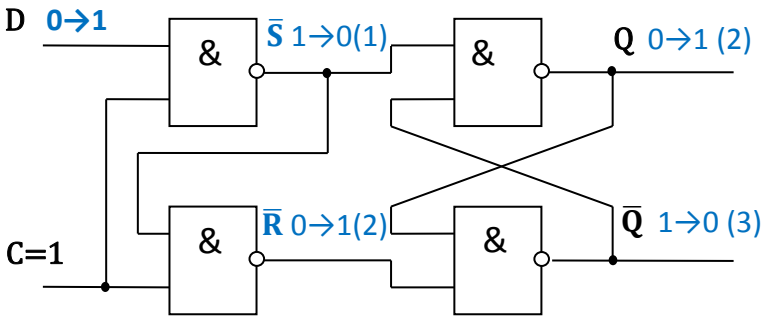


Рис. Г.2. Послідовність переключення елементів при зміні сигналу D в режимі наскрізного керування (C=1, тригер – відкритий для запису).

На рис. Г.3 показаний порядок поширення сигналів ***від зміни сигналу на вході C*** (0→1).

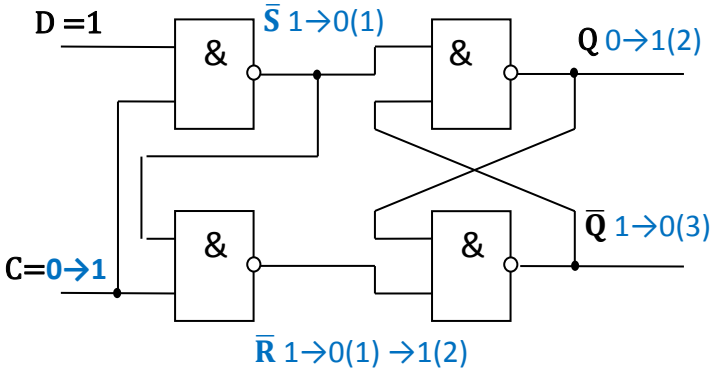


Рис. Г.3. Послідовність переключення елементів при зміні сигналу C в стандартному режимі (D переключився з 0 на 1 при C=0, коли тригер був закритий для запису)

Сигнал D змінився з 0 на 1 при C=0 (заборона запису). Потім C переключилося з 0 в 1, і почався процес запису. Маємо таку послідовність переключень: $\bar{S}$ і $\bar{R}$ переключаються одночасно, потім Q і ще раз $\bar{R}$, а потім $\bar{Q}$. Знову маємо Тзтр = 3 tзр.

Розглянемо більш детально порядок переключення елементів на рис. Г.2. Порядок переключення елементів визначається моментами часу Тпоточне + tзр, коли зміна сигналу на вході вентиля І-НЕ призводить до зміни сигналу на його виході.

- зміна D в момент T1 призводить до зміни сигналу nS після затримки T2=T1 + 1tзр,

- змінений $\overline{S}$ в момент T2 (T1 + 1tзр) призведе до перемикань $\overline{R}$ і Q в момент T3=T2+1tзр (T1 + 2tзр). Вони перемикаються паралельно, тому що почали з одного моменту часу T2 і мають однакові tзр.

- зміна $\overline{R}$ і Q викличе перемикання $\overline{Q}$ в T4= T3+1tзр (T1 + 3tзр).

Таким чином, разом від входу D (T1) до виходу $\overline{Q}$ (T4) маємо 3tзр.

Ось такий причинно - наслідковий зв'язок. Сигнал на виході елемента може змінитися тільки після зміни сигналу на його вході через tзр. Якщо в програмі моделювання tзр=0, то затримка вважається умовною і послідовність все одно зберігається.

ДЛЯ ТОГО, ЩОБ ПОБАЧИТИ ПОСЛІДОВНІСТЬ ПЕРЕМИКАННЯ ЕЛЕМЕНТІВ В СХЕМІ на часовій діаграмі потрібно присвоїти всім елементам однакові tзр, наприклад, по 5нс.

Г.1.2. Принцип роботи тригерів, керованих фронтом синхросигналу

При керуванні рівнем синхросигналу тригер реагує на зміни сигналу D на всьому інтервалі дозволу, що часто буває не зручно для побудови ЦП на їх основі. Мета побудови тригера, керованого фронтом, – зробити так, щоб він реагував на зміни сигналу D не на інтервалі дозволу, а тільки в момент його зміни (насправді, як побачимо далі, перед зміною).

Тригер складається з двох одноступеневих D тригерів – Master (Qm) і Slave (Qs). Сигнал синхронізації подається на тригери у протифазі (через інвертор). Схема приведена на рис. Г.4.

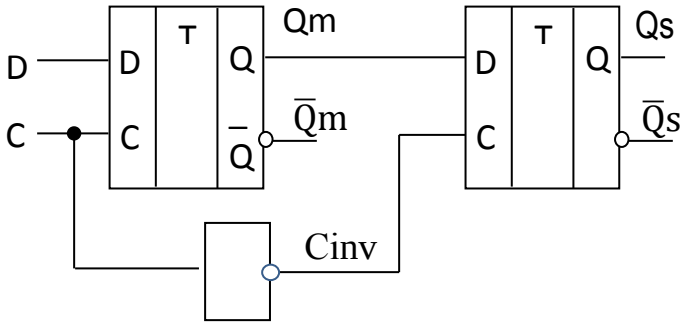


Рис. Г.4. Схема двохступеневого D-тригера Master-Slave Сінв

Завдяки інвертору Master і Slave працюють по-почергово – коли Qm записує нові дані, Qs– зберігає попередні дані. І навпаки, коли Qm переходить в режим зберігання, Qs переписує з нього нові дані.

До заднього фронту С запис у Qm дозволено, оскільки С=1, а запис у Qs заборонено, оскільки на виході інвертора 0. В цей час Qs зберігає старе значення. Після заднього фронту С Master переходить у режим зберігання останнього значення, яке він встиг записати при С=1, і перестає реагувати на зміни сигналу D. А Slave відкривається для запису (на виході інвертора 1) і переписує з Master нове значення, яке той зберігає.

На діаграмі (рис. Г.5) Qs до заднього фронту С зберігав 0.

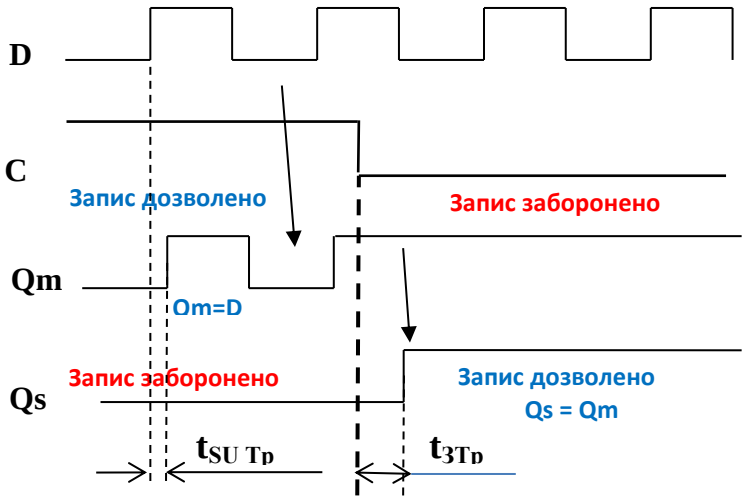


Рис. Г.5. Діаграма роботи двохступеневого D-тригера Master-Slave

На діаграмі можна побачити, що впродовж C=1 Qm змінилось кілька разів, а після переключення C в 0 – зберігає останнє записане значення 1. A Qs до заднього фронту C зберігав 0, бо був закритий для запису, а після фронту – відкрився для запису і переписав сигнал 1 з виходу Qm. Приклад вимірювання затримки D тригера показано на рис. Г.6.

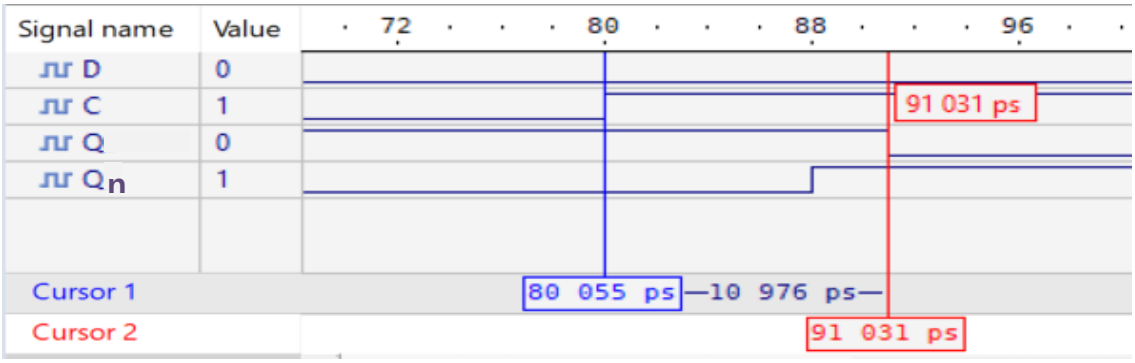


Рис. Г.6. Визначення Тзтр в моделі D тригера з синхронізацією за переднім фронтом

За цим принципом працюють всі базові схеми двохстпеневих тригерів, розглянутих з додатку В (M-S, 6-ти елементна, із забороною звя’зків). Тригери, побудовані на основі цих базових схем відрізняються тільки значеннями своїх параметрів.

Г.2. ДИНАМІЧНІ ПАРАМЕТРИ ТРИГЕРІВ

До основних динамічних параметрів тригерних схем відносять:

1. мінімальна і максимальна затримка тригера;
2. час передвстановлення та витримки інформаційних сигналів відносно сигналів синхронізації;
3. мінімально припустима тривалість інформаційних сигналів і сигналів асинхронного RS- встановлення;
4. мінімально припустима довжина такту синхронізації і мінімально припустима частота синхронізації.

Всі ці параметри можна виміряти по часовій діаграмі роботи тригера в різних режимах (перевіряється вся таблиця істинності). На діаграмах знаходиться максимальні значення відповідних параметрів.

Г.2.1. Затримка тригера

Затримка тригера є часом, який необхідний для переключення обох його виходів у новий стан з моменту, коли сигнал синхронізації С приймає дозволяюче значення.

$tз_{Тр} = max\ (tз_{Тр}{}^{01}{}_{max},\ tз_{Тр}{}^{10}{}_{max})$.

Одноступеневі тригери, керовані рівнем сигналу С

Як було розглянуть раніше ці тригери можуть працювати в двох режимах: *в асинхронному (або прозорому)* режимі і в режимі *однократного перемикання тригера* протягом такту синхросигналу,. В прозорому режимі тригер постійно відкритий для запису (на вход С подано дозволяючий рівень сигналу), і стан тригеру змінюється тільки від зміні інформаційних сигналів.

Для цих схем необхідно перевіряти затримки виходів для двох випадків: від зміни інформаційних сигналів при незмінному дозволяючому значенні С та від зміни сигналу С при незмінних інформаційних сигналах (рис.Г.2 та рис. Г.3). Потім потрібно вибирати максимальне з цих значень. Більшість схемних рішень реалізується так, що значення затримок тригера в обох випадках співпадають і можуть позначатись ще як час переключення тригера ($t_{пер}$).

Двохступеневі тригери, керовані фронтом сигналу С

Затримка цих схем вимірюється від дозволяючого фронту С до моменту встановлення останнього з виходів Q або nQ у новий стан. Ця затримка визначається затримкою у ланцюгу синхронізації і другого ступеня тригера Slave.

$tз_{Тр} = max\ (tз_{Тр}{}^{01}{}_{max},\ tз_{Тр}{}^{10}{}_{max})$.

Тобто, для схеми на рис.Г.4 затримка тригеру вимірюється від заднього фронту С до моменту встановлення Q і nQ у новий стан (рис.Г.5). В даному прикладі tзТр визначається затримкою у ланцюгу синхронізації

(tзр інвертора) та затримкою верхнього ступеня Slave (3tзр I-Hi). Таким чином, для схеми на рис. Г.4 $Тз_{Тр}$ =3tзр I-HI + tзр інвертора. При перемиканні С 1→ 0 (задній фронт) спочатку перемикається інвертор (tзр інвертора), після цього на вході 2-го ступеня з'являється дозволяючий рівень Cinv =1, далі починається запис з нижнього тригера в верхній тривалістю 3 tзр I-HI.

Для інших схем тригерів $Тз_{Тр}$ буде вимірюватись так само, але при цьому будуть переключатись інші елементи їх схем. Наприклад, для схем з «забороняючими зв’язками» і «6-елементної» без додаткових інверторів $Тз_{Тр}$ =3 tзр I-HE.

Г.2.2. Час передвстановлення і витримки інформаційних сигналів відносно сигналів синхронізації

З синхронізацією тригера пов'язано два важливі параметри — час передвстановлення t_{SU} (Set-Up Time) і час витримки t_H (Hold Time).

Час t_{SU} — це інтервал часу до надходження дозволяючого синхроімпульсу, протягом якого інформаційні сигнали повинні залишатися незмінними для надійного запису в тригер.

Час витримки t_H — це час після завершення дії дозволяючого синхроімпульсу, протягом якого інформаційний сигнал повинен залишатися незмінним. Цей час необхідний для того, щоб синхросигнал встиг заблокувати інформаційні входи тригера до того, як сигнали на них зміняться. Дотримання необхідних значень цих параметрів забезпечує правильне сприйняття тригером вхідної інформації.

Одноступеневі тригери, керовані рівнем сигналу С

Для цих тригерів при *роботі в прозорому режим* (асинхронне перемикання від зміни інформаційних сигналів на інтервалі дозволу С) поняття t_{SU} і t_H не має сенсу. Якщо передбачається тільки *однократне перемикання тригера* протягом такту синхросигналу, тоді t_{SU} – це час, необхідний на переключення тригера в режим запису, а t_H – в режим зберігання. Цей час необхідний для запобігання змагань, які можуть

виникнути в реальних схемах при одночасному перемиканні інформаційних сигналів та синхросигналу С. Через розкид затримок елементів в шляхах розповсюдження сигналів, тригер може відкритися раніше ніж інформаційні сигнали зміняться, або закритися пізніше зміни інформаційних сигналів. Щоб хибні дані не потрапляли в тригер, потрібно спочатку встановлювати потрібні дані, а потім відкривати тригер для їх запису, а після запису – спочатку закривати тригер (переводити в режим зберігання), а потім змінювати дані (рис. Г.7). В більшості схемних реалізацій час переключення тригеру в режим запису/зберігання даних сигналом С дорівнює затримці одного рівня вентилів.

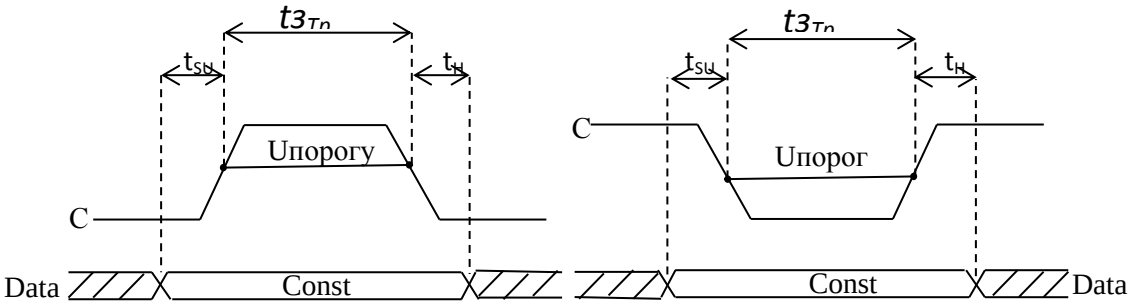


Рис. Г.7. Час передвстановлення і час витримки для тригерів, керованих рівнем одиниці і рівнем нуля синхросигналу С

Наприклад, для тригера на рис. Г.1, робота якого відповідно режиму однократного перемикання протягом такту С показана на рис. Г.3, $t_{SU} = t_H$, і вимірюється: t_H від моменту, коли С перемикається 1→0, до моменту, коли $\overline{R} = \overline{S} = 1$ (перехід в режим зберігання): t_{SU} – від моменту, коли С перемикається 0→1, до моменту першої зміни $\overline{R}$ або $\overline{S}$ з 1→0 (перехід з режиму зберігання в режим запису).

Двохступеневі тригери, керовані фронтом сигналу С

Для тригерів з динамічним управлінням час t_{SU} — це час, необхідний для надійного запису даних у нижній ступень тригера Master.

$t_{SU} = max\ (tзм^{01}_{max},\ tзм^{10}_{max})$.

Він вимірюється від моменту зміни інформаційних сигналів до моменту встановлення останнього з виходів Master у новий стан при незмінному дозволяючому наченні С.

Час t_H — це час після дозволяючого фронту С, який необхідний для того, щоб нижній ступень тригера Master переключився у режим зберігання і перестав реагувати на зміну вхідних сигналів.

Часові параметри t_{SU} і t_H показано на рис. Г.8.

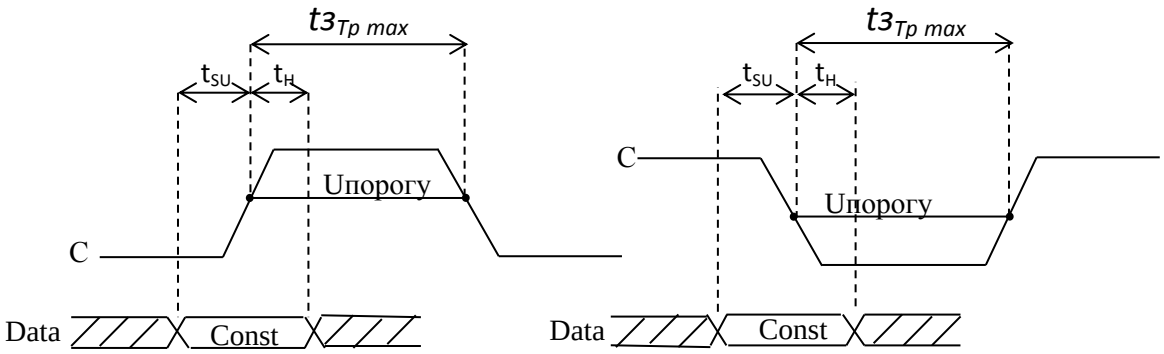


Рис. Г.8. Час передвстановлення і час витримки для тригерів, керованих переднім і заднім фронтом синхросигналу С

Фактично час t_{SU} – це максимальна затримка одноступеневого тригеру Master, і вимірюється вона так само, як і затримка одноступеневого тригеру (рис. Г.5). Для тригера з рис. Г.4 t_{SU} =3 tзр I-HI.

Час t_H для ступеня тригеру Master має той же сенс, що і для одноступеневого тригеру в режимі однократного перемикання протягом такту синхросигналу, тобто – це час, необхідний на перемикання Master в в режим зберігання. Його можна виміряти від дозволяючого фронту С до моменту переключення тих елементів ступеня Master, після переключення яких він перейде в режим зберігання, а інформаційні сигнали заблокуються і перестануть впливати на роботу схеми.

Для тригера з рис. Г.4 вимірюється від заднього фронту С до моменту встановлення на обох входах елементу пам’яті Master сигналів, відповідних режиму зберігання $\overline{R} = \overline{S} = 1$. Відповідно до рис.Г.1, як тільки С перемикнеться з 1→0, $\overline{R}$ і $\overline{S}$ одночасно переключаться в 1: nS= $\overline{(D\&C)}$= $\overline{(D\&0)}$=1 та nR= $\overline{(S\&C)}$= (S&0)=1. Після цього сигнал D блокується С=0 і втрачає вплив на схему, а комбінація $\overline{R} = \overline{S} = 1$ переводить комірку пам’яті Master (Qm і $\overline{Qm}$) в режим зберігання. Таким чином, t_h =1 tзр I-HI.

Для інших тригерів з динамічним управлінням t_h і t_{SU} має той самий сенс. Тільки процес переключення нижнього ступеня і процес блокування інформаційних входів після дозволяючого фронту може забезпечуватись по різному. В кожному випадку потрібно аналізувати конкретну схему.

У схем MS із «забороняючими зв'язками» принцип роботи аналогічний 2-х ступеневій MS схемі з інвертором. Тільки функції, які виконуються завдяки інвертору в цій схемі виконують забороняючі зв'язки A та B на D5 та D6.

А ось «6-елементну» схему (рис.Г.9) розглянемо більш докладно.

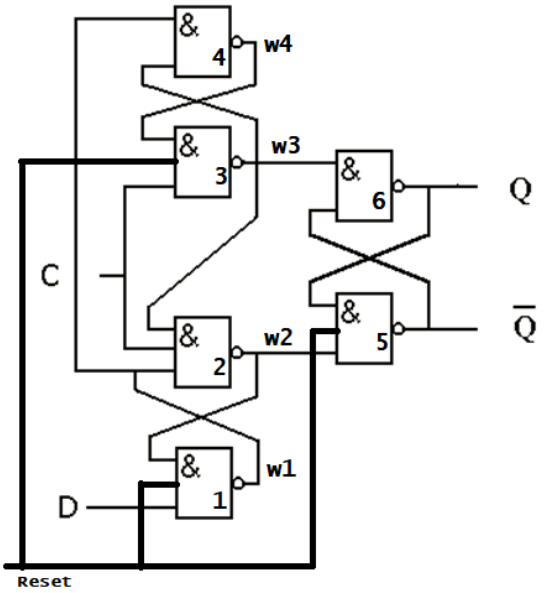


Рис. Г.9. Схема двохступеневого D-тригера «6-елементна»

Схема D-тригера на рис.Г.9 переключається за переднім фронтом C. При C=0 елементи 2 і 3 перемикаються в «1», що забезпечує стан зберігання для верхнього тригера на елементах 5, 6 (Q та $\overline{Q}$), а сигнал D перемикає виходи елементів 1 і 4: W1= $\overline{W2\&D} = \overline{1\&D} = \overline{D}$ та аналогічно W4=D. Після переднього фронту C (0->1) відбувається розблокування (переключення) елементів 2 і 3, в результаті чого стає W2= D і W3= $\overline{D}$. Тепер нове значення D через W2 і W3 запишеться в верхній тригер (Q та $\overline{Q}$).

Після переднього фронту C (при C=1) нижня частина схеми переходить у режим зберігання, який забезпечується зворотними зв'язками на елементах 1 та 2, елементах 3 та 4 і елементах 3 та 2 разом з блокуванням входу D. Блокування входу D забезпечується або сигналом «0» з елементу 2 (елемент 1 утримується в «1»), або сигналом «0» з елементу 3 (елементи 2 і 4 утримуються в «1»). Це залежить від даних, записаних до фронту (при C=0) на елементи 1 і 4 під час Tsu. Тепер зміна сигналу D вже не зможе через елементи 1 і 4 змінити стан елементів 2 і 3 (забезпечується режим зберігання в нижній частині схемі).

Тому в цьому тригері th достатньо поміряти від переднього (дозволячого) фронту до перемикання елементу 2 або 3 (1 tзр I-HI). А Tsu вимірюється від зміни D, або від зміни C на 0 (як було описано раніше) до моменту перемикання останнього з елементів 1 та 4 (2 tзр I-HI). Слід зауважити, що після перемикання C в 0 також буде спостерігатись переключаться в «1» на елементи 2, 3 для забезпечення режиму зберігання у верхньому тригері (на елементах 5, 6). Але до Tsu (скільки потрібно утримувати D до фронту для його надійного запису у нижню тригерну схему) це відношення не має.

Коли може бути th=0

Якщо б ми схотіли створити, наприклад, двохступеневий тригер, з входом $\overline{D}$, то його частина Master виглядала б як на рис.Г.10.

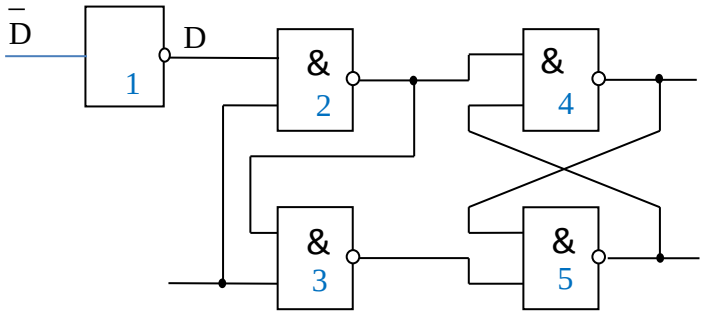


Рис. Г.10. Схема частини Master для $\overline{D}$тригера.

В такого тригера th=0. Тому, що при одночасній зміні сигналів на обох входах $\overline{D}$ і С завжди гарантується, що С переключить елементи 2 і 3 швидше ніж $\overline{D}$, якому спочатку треба ще переключити елемент 1. А ось tsu такого тригера збільшиться на Тзр елементу 1.

Г.2.3. Мінімально припустима тривалість вхідних сигналів

Мінімально припустима тривалість вхідних сигналів є часом, протягом якого слід утримувати відповідні сигнали незмінними для того, щоб вони встигли надійно записатися в тригер. Параметр позначається як $t_{вх.min}$. В синхронних тригерах зазвичай визначають мінімально припустима *тривалість інформаційних сигналів* (D, JK, RS…) та *сигналів асинхронного RS- встановлення* для комбінованих тригерів з синхронізацією за фронтом С. В тригерах з синхронізацією за фронтом входи асинхронного встановлення використовуються для швидкого встановлення тригера в певний стан, оскільки синхронний запис відбувається в них значно довше. Для одноступеневих тригерів RS- встановлення практично не застосовується.

Мінімально припустима тривалість сигналів асинхронного RS-встановлення $t_{RS.min}$ в комбінованих тригерах має бути достатньою для переключення обох виходів тригера у бажаний стан, і вимірюється як найбільша з затримок від моменту зміни R або S сигналу до переключення останнього з виходів тригера (Q і $\overline{Q}$) .

Одноступеневі тригери, керовані рівнем сигналу С

Для одноступеневих схем, керованих рівнем сигналу С, тривалість інформаційних сигналів має бути не меншою ніж максимальна затримка тригера: $t_{вх.min}$. = $tз_{Тр}$. А *в режимі однократного запису* вхідної інформації протягом дії дозволяючого рівня синхроімпульсу значення інформаційних сигналів не повинне змінюватись впродовж всього інтервалу, поки запис в тригер дозволений, $t_{вх.min}$= t_{SU} + $tз_{Тр}$ + t_H (рис. Г.7).

Двохступеневі тригери, керовані фронтом сигналу С

Для двохступеневих схем, керованих фронтом сигналу С, тривалість інформаційних сигналів має бути: $t_{вх.min} = t_{SU} + t_H$ (рис. Г.8).

Для схеми двохстпеневого Master-Slave D тригера з рис. Г.4 асинхронне RS встановлення може бути зорганізовано так само, як для схеми MS тригера із забороняючими зв'язками на рис.Г.11.

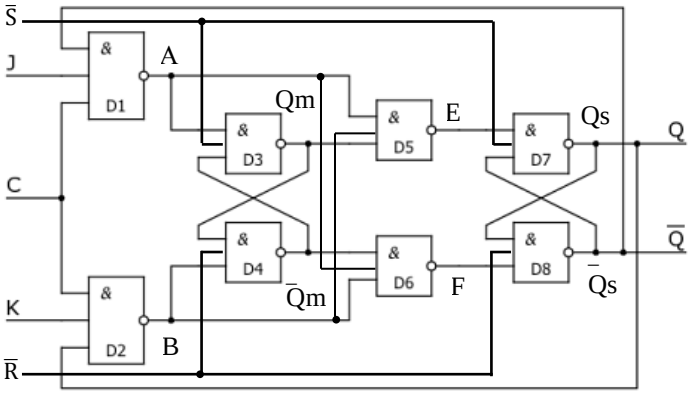


Рис. Г.11. Схема двохступеневого JK-тригера «із забороняючими зв'язками»

Комірки пам'яті обох ступеней (Qm і $\overline{Q}$m та Qs і $\overline{Q}$s) тригера встановлюються паралельно. А, оскільки затримка комірки пам'яті дорівнює 2 tзр, то $t_{RS.min}$= 2tзр І-НІ.

Організація асинхронного RS встановлення для «6-елементної» схеми D-тригера показана на рис. Г.9. Варто відзначити, що для цієї схеми $\overline{R}$ на відміну від інших схем MS тригерів повністю домінує над сигналами С і D, і забезпечує встановлення нижнього ступеня (елементи 1 і 4) і верхнього ступеня (Q/5 і $\overline{Q}$/6) при будь-яких значеннях на входах С і D за 2 tзр І-НІ. Тоді як для інших схем MS (2-х ступіневої та з забороняючими зв'язками) при певних комбінаціях значень на інших входах можливі збої. Хоча такі ситуації можуть виникати достатньо рідко, а схемне рішення спрощується.

Г.2.4. Мінімально припустима частота синхронізації

Мінімально припустима частота синхронізації вираховується, виходячи з мінімально припустимої довжини такту синхронізації, який має бути достатнім для забезпечення розглянутих часових параметрів тригера. Такт С (рис.Г.12) складається з напівперіодів одиниці і нуля T_{min} =T0+T1.

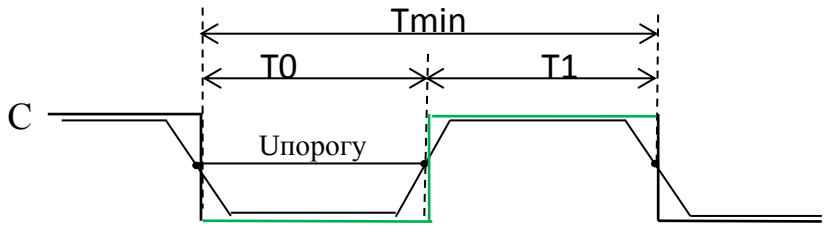


Рис.Г.12. Параметри такту синхронізації С

Тоді мінімально припустима частота синхронізації:

$f_{Tpmax} = 1/T_{min}$.

Одноступеневі тригери, керовані рівнем сигналу С

Для забезпечення роботи тригера в режимі однократного перемикання протягом такту синхроімпульсу його період згідно рис.Г.7 і рис.Г.12 повинен бути:

$T_{min} \geq max\ (tз_{Tp}{}^{01}{}_{max},\ tз_{Tp}{}^{10}{}_{max}) + t_{SU} + t_H$.

При цьому буде T1≠ T0.

Для забезпечення роботи тригера *в прозорому режимі* висувається вимога тільки до довжини дозволяючого напівперіоду, який повинен бути не менше, ніж максимальна затримка тригера $tз_{Tp}=max\ (tз_{Tp}{}^{01}{}_{max},\ tз_{Tp}{}^{10}{}_{max})$. А оскільки $tз_{Tp}$ зазвичай > t_{SU} +t_H, то при симетричному такті синхросигналу (T0=T1) для забезпечення роботи тригера в обох режимах:

$T_{min} \geq 2max\ (tз_{Tp}{}^{01}{}_{max},\ tз_{Tp}{}^{10}{}_{max})$.

а частота синхросигналу синхросигналу:

$f_{Tpmax} \leq 1/2\ tз_{Tp\ max}$.

Двохступеневі тригери, керовані фронтом сигналу С

Для тригерів, які переключаються за фронтом синхросигналу, відповідно до рис. Г.12 та рис. Г.8:

$T_{min} \geq t_{SU} + max\ (tз_{Tpmax}, t_{H\ max})$.

Оскільки $tз_{Tp}$ завжди > t_H, то:

$T_{min} \geq t_{SU} + max\ (tз_{Tp}{}^{01}{}_{max},\ tз_{Tp}{}^{10}{}_{max})$.

Ця величина може позначатись ще як час розповсюдження сигналу через тригер ($t_{роз}$).

Довжин півперіодів Т1 і Т0 повинна бути достатньою для переключення відповідних частин тригера. Якщо синхроімпульс повинен бути симетричним (Т0 = Т1), то:

$f_{Tpmax} \leq 1/2\ max\ (tз_{Tp\ max}, t_{SU})$.

У прикладі D-тригера з синхронізацією по задньому фронту (рис. Г.4) для надійного перемикання ступеня Master потрібно, щоб протягом часу t_{SUmax} на вході С утримувалась «1», а щоб надійно перемикався ступень Slave та закривався Master потрібно, щоб протягом часу $max(tз_{Tpmax}, t_H)$ на вході С утримувався «0». Таким чином:

$T_{min} = T1 + T0 \geq t_{SU\ max} + tз_{Tp\ max}$,

T_{min} = (3 tзр І-НІ) + (3 tзр І- НІ + tзр інвертора).

Оскільки зазвичай у синхросигнала напівперіоди повинні бути однакові Т1 = Т0, то:

$T_{min} \geq 2 * max\ (t_{SU\ max,}\ tз_{Tp\ max})$.

В даному випадку $tз_{Tp\ max} > t_{SU\ max}$, тому:

$T_{min} \geq 2*$ (3 tзр І-Ні + tзр інвертора).

$f_{Tpmax} \leq 1/T_{min}$.

ДОДАТОК Д.
КОРОТКІ ВІДОМОСТІ ПРО РЕГІСТРИ І ЛІЧИЛЬНИКИ

Д.1. РЕГІСТРИ

Регістрами називаються послідовні цифрові пристрої, що виконують функції прийому, зберігання і передачі інформації.

Вони оперують з багато розрядними двійковими числами, які також називають словами. Над словами виконується ряд операцій:

- Прийом слова у прямому і інверсному коді;
- Видача слова в прямому і інверсному коді;
- Виконання порозрядних логічних операцій над словами;
- Порозрядний зсув коду вправо чи вліво.

Регістри будуються з розрядних схем, які складаються з тригерів і, найчастіше, додаткових логічних елементів.

Головною класифікаційною ознакою регістрів є спосіб прийому і видачі даних. За цією ознакою розрізняють *паралельні* (статичні) регістри, *послідовні* (що зрушують) і *паралельно-послідовні*.

У паралельних регістрах прийом і видача слів проводяться по всіх розрядах одночасно. У них зберігаються слова, які можуть бути піддані порозрядним логічним перетворенням.

У послідовних регістрах слова приймаються і видаються послідовно розряд за розрядом. Їх називають регістрами зсуву оскільки тактуючі сигнали при введенні і виведенні слів переміщають їх на один розряд у розрядній сітці. Регістр може здійснювати зсув записаного слова вліво, управо, або бути реверсивним (з можливістю зсуву в обох напрямах).

Послідовно-паралельні регістри мають входи-виходи одночасно послідовного і паралельного типу. Є варіанти з послідовним входом і паралельним виходом, паралельним входом і послідовним виходом, а

також варіанти з можливістю будь-якого поєднання способів прийому і видачі слів.

Паралельний регістр використовується для короткочасного зберігання чисел в паралельному двійковому коді. Тому паралельні регістри називаються ще регістрами пам'яті.

У паралельному регістрі код числа, що запам'ятовується, подається на інформаційні входи всіх тригерів і записується в регістр із приходом тактового імпульсу. Вихідна інформація змінюється з подачею нового вхідного слова і приходом наступного імпульсу запису.

Найчастіше паралельні регістри будуються на D-тригерах як з динамічною (за фронтом синхроімпульсу), так і з статичною (за рівнем синхроімпульсу) синхронізацією.

Загальними для всіх розрядів зазвичай є ланцюги синхронізації, скидання/установки, та ланцюги управління. Приклад схеми паралельного регістра, побудованого на тригерах типу D з прямим динамічним управлінням C, входом скидання R і виходами с третім станом, які управляються сигналом E, показаний на рис. Д.1.

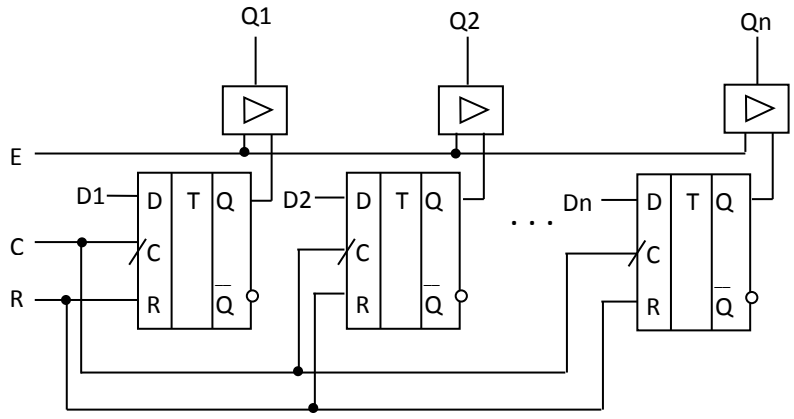


Рис. Д.1. Схема паралельного регістра на D-тригерах з динамічним управлінням C, входом скидання R і виходами с третім станом.

Таблиця істинності регістра, показаного на рис. Д.1, надана у таб. Д.1.

Таблиця Д.1. Таблиця істинності регістра (рис. Д.1)

Режим	**Входи**				**Виходи**				
	C	R	E	Di	Q0	Q1	. . . .	Q6	Q7
Скидання	–	1	1	–	0	0	. . . .	0	0
Зберігання	↓,0,1	0	1	–	Q0	Q1	. . . .	Q6	Q7
Паралельне завантаження	↑	0	1	0/1	D0	D1	. . . .	D6	D7
Високий імпеданс	–	–	0	–	Z	Z	. . . .	Z	Z

Послідовний регістр призначений для короткочасного зберігання інформації, але, на відміну від паралельного регістра, в ньому здійснюється логічна операція зсуву коду числа, що зберігається, на будь-яку кількість розрядів. Введення інформації в послідовний регістр здійснюється по послідовному входу D. Зсув коду числа відбувається за допомогою синхронізуючих імпульсів C, в результаті подачі яких здійснюється зсув всіх розрядів числа з входу до виходу або навпаки.

Послідовні регістри найчастіше будуються на D-тригерах, але на відміну від паралельних регістрів використовуються тільки тригери з динамічним управлінням (рис. Д.2). Це гарантує, що впродовж одного такту синхронізації всі тригера переключаться тільки один раз.

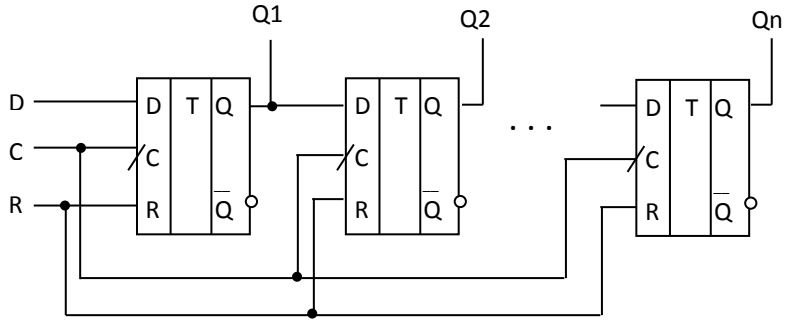


Рис. Д.2. Схема послідовно - паралельного регістра на D-тригерах з динамічним управлінням C і входом скидання R

З першим синхроімпульсом біт інформації з входу D записується у перший тригер, з виходу першого тригера – у другий тригер, з другого – у третій і так далі. З приходом другого синхроімпульсу у перший тригер запишеться нове значення з входу D. Таким чином з приходом кожного синхроімпульсу відбувається зсув інформації на один розряд управо. Тому послідовні регістри називають ще регістрами зсуву. Для організації зсуву уліво, необхідно послідовно з'єднати вихід останнього тригера n зі входом n-1-го тригера, вихід n-1-го – зі входом n-2-го і т.п., а вхід D підключити до входу тригера n. У послідовних регістрах інформація виводиться з виходу Q останнього (зсув вправо) або першого (зсув вліво) тригера.

Широкого розповсюдження набули ***послідовно-паралельні*** регістри, які мають паралельне виведення інформації. Для цього в послідовному регістрі достатньо використовувати виходи Qi всіх тригерів. Такі регістри можна використовувати в якості перетворювачів послідовного коду в паралельний.

Якщо вихід останнього тригера з'єднати з входом першого, то вийде ***кільцевий регістр зсуву.*** Записана в його розряди інформація під впливом синхроімпульсів циркулюватиме по замкнутому кільцю.

Д.2. ЛІЧИЛЬНИКИ

Лічильниками називаються послідовні цифрові пристрої, що виконують функції підрахунку кількості вхідних імпульсів і зберігання результату рахунку до приходу наступного імпульсу, коли рахунок зміниться.

Велика різноманітність типів лічильників викликана їхнім широким використанням як в обчислювальній техніці, так і в різних пристроях автоматики. Вони застосовуються для утворення послідовностей адрес команд, для підрахунку числа циклів виконання операцій, для запам'ятовування коду в аналого-цифрових перетворювачах і т.п.

За способом кодування внутрішніх станів розрізняють двійкові лічильники, які рахують у коді 8421, і лічильники, які рахують у інших

кодових системах, наприклад, лічильник з кодом Джонсона, лічильники з кодом "1 з N" і ін. Найбільше поширення мають двійкові лічильники.

За напрямом рахунку лічильники діляться на ті, що складають (прямого рахунку), ті, що віднімають (зворотного рахунку) і реверсивні (із зміною напряму рахунку).

За приналежності до певного класу автоматів говорять про синхронні або асинхронні лічильники.

Можливі режими роботи лічильника:

- підрахунок числа вхідних сигналів;
- ділення частоти.

У першому режимі результатом є код, що формується на виходах лічильника, в другому режимі – послідовність сигналів з певного виходу лічильника із частотою у К разів меншою за частоту послідовності вхідних сигналів.

Основними параметрами лічильника є:

- модуль рахунку або коефіцієнт перерахунку М;
- швидкодія лічильника.

Модуль визначає число можливих станів лічильника. Після надходження на лічильник М вхідних сигналів, лічильник повертається в початковий стан і починає рахувати спочатку. Такі лічильники називаються також дільниками на число, рівне М.

Швидкодія лічильника визначається двома величинами:

- часом встановлення $t_{вст}$ лічильника, тобто інтервалом часу між моментом надходження вхідного сигналу і моментом завершення переходу лічильника в новий стійкий стан, тобто встановленням на його виходах нового коду;
- максимальною частотою вхідних сигналів f_{max}. При цьому повинна виконуватися умова $f_{max} < 1/$ твст.

Лічильники будуються як ланцюжок послідовно включених тригерів, що мають міжрозрядні зв'язки. Відповідно організації цих зв'язків

розрізняють лічильники з послідовним, паралельним і комбінованим перенесенням.

Лічильники найзручніше будувати на тригерах типу Т (рахункові) і JK, які при J = К = 1 працюють у рахунковому режимі. З тих же причин, що і в регістрах зсуву, тригери повинні мати динамічне управління.

Стан лічильника читається по виходах розрядних схем як слово Qn-1, Qn-2, ... Q0, вхідні сигнали поступають на молодший розряд лічильника, який знаходиться ліворуч. Розрядність лічильника, а отже і число тригерів визначаються максимальним числом, до якого він повинний рахувати у заданій системі кодування.

Двійкові лічильники

Двійкові лічильники це лічильники, що мають модуль $M = 2^n$, де n — ціле число, і природну послідовність кодування станів (його стани відображаються послідовністю двійкових чисел, десятковими еквівалентами яких будуть числа 0, 1, 2, 3..., M—l).

Кількість тригерів для двійкових лічильників визначається формулою:

$$N=\log_2 M.$$

Найбільш зручним Для побудови є тригер Т-типа (рахунковий тригер), який здійснює підрахунок імпульсів по модулю 2, тобто по суті є простим лічильником з *M=2*. З'єднавши декілька рахункових тригерів чином, можна отримати схему багаторозрядного лічильника.

Асинхронний лічильник прямого рахунку (що складає) на базі Т-тригерів з інверсним динамічним управлінням показано на рис. Д.3.

Для виконання операції віднімання для схеми досить змінити зв'язки з прямих виходів Q тригерів на виходи Т (рис. Д.3(а)) на зворотні – з виходів $\overline{Q}$ на виходи Т (рис. Д.3(б)). Тоді, в ті моменти часу, коли на діаграмах Qi буде спостерігатися передній фронт сигналу, на виходах $\overline{Q}$ і буде – задній фронт, від якого і буде переключатися наступний тригер.

Таким чином, шляхом перемикання виходів з інверсних на прямі і назад можна отримати лічильники, що як складають, так і віднімають.

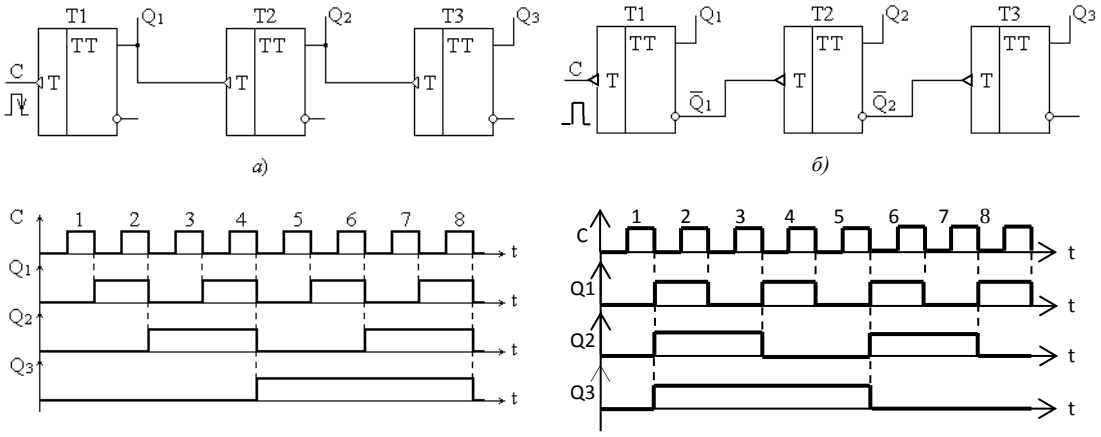


Рис. Д.3. Структурна схема і часова діаграма роботи трьохрозрядного двійкового лічильника на основі Т-тригерів з інверсним динамічним управлінням: а) прямого рахунку, б) зворотного рахунку

Якщо використовувати тригери з прямим динамічним управлінням, то сигнали на входи подальших тригерів, в порівнянні зі схемою рис. Д.3. необхідно подавати навпаки: для складання – з інверсних виходів $\overline{Q}$, для віднімання – з прямих виходів Q. Аналогічно, для схем на рис Д.4 при використанні тригерів з інверсним динамічним управлінням для операції додавання одиниці необхідно подавати сигнали з інверсних виходів $\overline{Q}$ на входи Т подальших тригерів, а для операції віднімання одиниці – з прямих виходів Q.

Для асинхронних лічильників у найгіршому випадку перенесення розповсюджується по всій розрядній сітці від молодшого розряду до старшого, тобто для встановлення нового стану повинні перемкнутися послідовно всі тригери. Звідси видно, що час встановлення коду складає:

$t_{вст} \leq nt_{зтг}$, де $t_{зтг}$ – затримка одного тригера. Зі збільшенням розрядності лічильника його сумарна затримка зростає, припустима частота вхідних сигналів f_{max} зменшується.

Для усунення цього недоліку використовуються лічильники, у яких всі тригери спрацьовують одночасно. Такі лічильники отримали назву ***синхронних лічильників***.

Ідея синхронного лічильника полягає в побудові комбінаційної схеми (на елементах І), що паралельно формує сигнали перенесення на всі розряди, згідно яким відбуватиметься одночасне перемикання тільки частини тригерів у залежності від поточного вихідного коду (рис. Д.4).

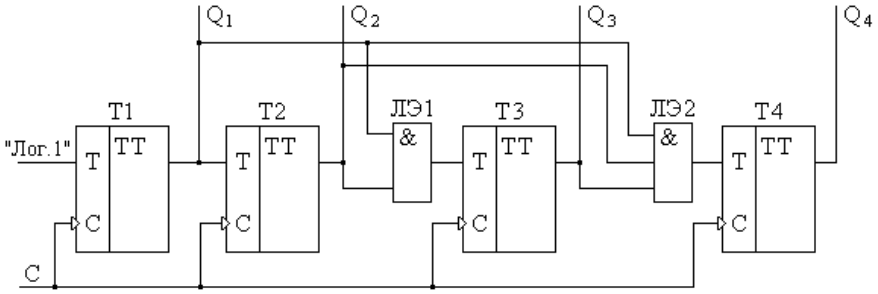


Рис. Д.4. Структурна схема чотирьох розрядного лічильника прямого рахунку з паралельним перенесенням.

Щоб отримати схему піднімаючого лічильника, необхідно використовувати не прямі, а інверсні виходи тригерів.

Час встановлення таких лічильників не залежить від розрядності n і дорівнює $t_{вст} = t_{зтг}, + t_{з\,І}$, де $t_{з\,І}$ — затримка елемента І.

Недоліком лічильників з паралельним перенесенням є необхідність використання у разі збільшення розрядності лічильника елементів І з великим числом входів. При цьому виходи тригерів повинні мати високу здатність до навантаження.

Інший варіант структури синхронного лічильника – це структура **з *наскрізним перенесенням.*** Згідно цій структурі, перенесення формується тільки з одиничних результатів сусідніх розрядів (рис. Д. 5).

До подачі нового вхідного сигналу потрібно дати ланцюжку вентилів встановитися в новий стан шляхом їх послідовного перемикання. Тому час

встановлення таких лічильників дорівнює $t_{вст} \le t_{зтг}, + n\ t_{з\ I}$, де $t_{з\ I}$ — затримка елемента І.

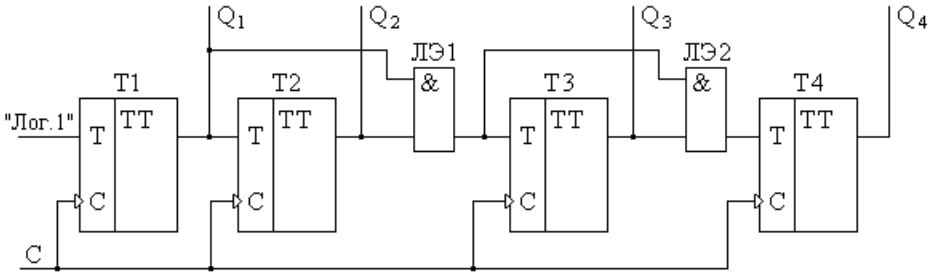


Рис. Д.5. Структурна схема чотирьох розрядного лічильника прямого рахунку з наскрізним перенесенням.

Двійкові лічильники *з довільним коефіцієнтом рахунку*. Часто потрібні лічильники із числом стійких станів, відмінних від 2^n. Наприклад, в електронних годинниках є мікросхеми з коефіцієнтами рахунку 6 (десятки хвилин), 10 (одиниці хвилин), 7 (дні тижня), 24 (години). Для побудови лічильника з М не рівним 2^n можна використовувати двійковий лічильник з N тригерів при умові $2^{n-1}<M<2^{n.}$. Такий лічильник буде мати зайві стійкі стани (2N-M). Виключити непотрібні стани можна використанням зворотних зв'язків, по ланцюгах яких лічильник перемикається в нульовий стан у тому такті роботи, у якому він дораховує до числа М.

Наприклад, для лічильника з К =10 потрібний чотирьох розрядний двійковий лічильник (тому що $2^3<10<2^4$). У тому такті, коли він мав би перейти в стан, відповідний числу 10, його необхідно знову встановити у нульовий стан. Для цього необхідно завести виходи розрядів, що відповідають числу 10 («1010»), на вхід елементу І (в даному випадку достатньо врахувати (Q2&Q4), тому що від 0000 до 1010 такої комбінації більше не зустрічається), а вихід елементу І – на входи R–встановлення всіх тригерів в «0». На самому початку одинадцятого стану (число 10) на виході елемента І з’явиться сигнал «1» скидання всіх тригерів лічильника в нульовий стан.

Розглянутий лічильник є двійковим еквівалентом рахункової декади, тому він називають двійково-десятковим.

Д.3. БАГАТОФУНКЦІОНАЛЬНІ РЕГІСТРИ І ЛІЧИЛЬНИКИ

Універсальні схеми можуть мати додаткові входи керування вибором режиму роботи схеми, що дозволяє виконувати різні операції. Універсальні регістри можуть поєднувати у собі функції паралельних і послідовних регістрів: паралельне завантаження, скидання в «0», встановлення у «1», зсув управо, зсув уліво, видача коду у прямому або інверсному вигляді, переключення виходів у високоімпедансний стан. Приклад таблиці істинності універсального регістра, що має вхід асинхронного скидання $\overline{R}$, входи So і S1 вибору синхронного (за переднім фронтом C) режиму: паралельне завантаження коду з входів Di, послідовний зсуву управо з входу DSR, послідовний зсуву уліво з входу DSL, показано в таб. Д.2. Рискою «–» у таблиці позначено входи, які не впливають на роботу схеми у певному режимі, тому значення сигналів на них не мають значення.

Таблиця Д.2. Таблиця істинності універсального регістра

Режим	**Входи**							**Виходи**				
	C	**$\overline{R}$**	**SO**	**S1**	**DSR**	**DSL**	**Di**	**Q0**	**Q1**	**. . . .**	**Q6**	**Q7**
Скидання	**–**	**0**	**–**	**–**	**–**	**–**	**–**	**0**	**0**	**. . . .**	**0**	**0**
Зберігання	**↑**	**1**	**0**	**0**	**–**	**–**	**–**	**Q0’**	**Q1’**	**. . . .**	**Q6’**	**Q7’**
Зсув уліво	**↑**	**1**	**1**	**0**	**–**	**0/1**	**–**	**Q1’**	**Q2’**	**. . . .**	**Q7’**	**DSL**
Зсув управо	**↑**	**1**	**0**	**1**	**0/1**	**–**	**–**	**DSR**	**Q0’**	**. . . .**	**Q5’**	**Q6’**
Паралельне завантаження	**↑**	**1**	**1**	**1**	**–**	**–**	**0/1**	**D0**	**D1**	**. . . .**	**D6**	**D7**
Зберігання	**↓,0, 1**	**1**	**–**	**–**	**–**	**–**	**–**	**Q0’**	**Q1’**	**. . . .**	**Q6’**	**Q7’**

Універсальні лічильники можуть поєднувати у собі функції паралельних регістрів і лічильників: скидання/встановлення, паралельне

завантаження, складання і віднімання, представлення вихідного коду у різному вигляді.

Приклад таблиці істинності реверсивного лічильника, що має вхід асинхронного скидання $\overline{R}$ і вхід S вибору синхронного (за заднім фронтом C) режиму складання або віднімання, показано в таб. Д.3. Рискою «–» у таблиці позначено байдужі значення сигналів, які не впливають на роботу схеми у даному режимі.

Таблиця Д.3. Таблиця істинності реверсивного лічильника

Режим	**Входи**			**Виходи**				
	C	$\overline{\mathbf{R}}$	**SO**	**Q0**	**Q1**	**. . . .**	**Q6**	**Q7**
Скидання	**–**	**0**	**–**	**0**	**0**	**. . . .**	**0**	**0**
Зберігання	**↑,0,1**	**1**	**–**	**Q0'**	**Q1'**	**. . . .**	**Q6'**	**Q7'**
Інкримент	**↓**	**1**	**1**	**Q7'. . . . Q1'Q0' + 1**				
Декримент	**↓**	**1**	**0**	**Q7'. . . . Q1'Q0' – 1**				

Функція вибору режиму за значенням сигналу на керуючому вході реалізується на основі елементарного мультиплексору (рис. Д.6.).

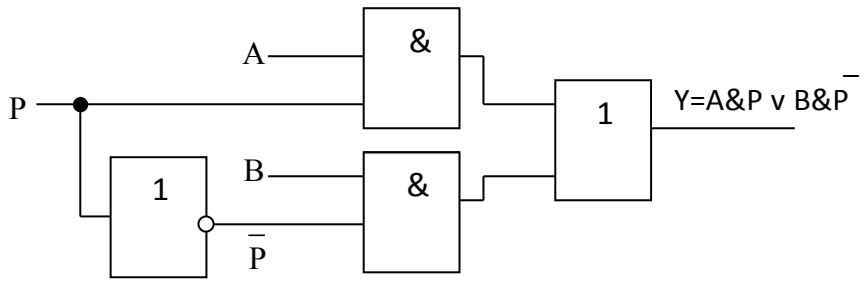


Рис. Д.6. Схема вибору одного з 2-х сигналів A чи B по значенню керуючого сигналу P: якщо P=1, Y=A v 0=A; якщо P=0, Y=0 v B=B

Наприклад, для реалізації реверсивних пристроїв таку схему встановлюють на вході кожного тригерного розряду. Тоді, для реалізації зсуву управо (E=1) чи уліво (E=0) у реверсивному регістрі вона буде

виконувати функцію $Di = EQ_{i-1} \vee \overline{E}\, Q_{i+1}$; для реалізації рахунку у прямому (E=1) чи у зворотному (E=0) напрямі у реверсивному лічильнику, побудованому аналогічно рис. Д.3, треба реалізувати функцію $Ti = EQ_{i-1} \vee \overline{E}\, \overline{Q}_{i-1}$. Для кільцевого регістру (E=0) з послідовним завантаженням і зсувом управо (E=1) таку схему можна встановити на вході 1-го розряду і вона буде виконувати функцію: $Di = EDR \vee \overline{E}\, Q_n$ (вибір сигналу з останнього розряду або з зовнішнього входу DR).

ДОДАТОК Е
КОРОТКІ ВІДОМОСТІ ПРО ПАКЕТИ АРИФМЕТИЧНИХ ФУНКЦІЙ

VHDL – це мова зі строгою типізацією даних. Функції, описані в пакеті std_logic_1164, не дозволяють спільно використовувати в операціях типи std_logic, integer та natural. Для проектування пристроїв, які використовують арифметичні операції, до бібліотеки IEEE в різний час було додано ряд пакетів. Пакети numeric_std та numeric_bit, які були розроблені пізніше, затверджені як стандарт. Їх функції підтримуються всіма синтезаторами і найбільш раціонально відображаються в апаратурі. Функції цих пакетів перезавантажують (замінюються при викликах) ряд стандартних операцій та функцій над цілими числами, а також однойменні операції та функції попередніх пакетів std_logic_arith, std_logic_signed std_logic_unsigned.

Пакети Numeric_std та Numeric_bit

Ці пакети відрізняються між собою тим, що функції та типи оголошені над різними базовими типами даних: std_logic та bit відповідно.

У пакетах визначено типи бітових векторів: *unsigned* – позитивні цілі двійкові числа без знака та *signed* – цілі двійкові числа, у яких старший біт вважається знаковим («0» – позитивне число, «1» – негативне число). Також мається на увазі, що негативні числа представлені у додатковому коді. Нагадаємо, що додатковий код числа отримується інвертуванням всіх розрядів його прямого коду і додаванням одиниці в молодший розряд результату. Приклади знакових чисел: +3 = 011, -3 = 101. При проектуванні апаратури необхідно пам'ятати, що довжина знакових чисел має бути на 1 (старший) розряд більше ніж беззнакових.

У пакеті визначені логічні функції, які перезавантажуються, (not, and, or, nand, nor, xor, xnor), арифметичні функції (+, -, *, mod, rem, abs), функції зсуву (srl, sll, ror, rol), а також функції порівняння для типів signed, unsigned та integer у різних поєднаннях.

Слід зазначити, що арифметичні, логічні та зсувні операції над типами signed і unsigned побітно виконуються однаково. Різниця проявляється в інтерпретації чисел (розширення розрядності, порівняння, перетворення на integer). При відніманні число, що віднімається, автоматично перетворюється на додатковий код.

В арифметичних операціях не можна змішувати операнди типу signed та unsigned. Також не можна привласнювати тип signed типу unsigned і навпаки. Однак, один з операндів може бути типу signed або unsigned, а інший – типу integer (або його підтипу).

Довжина числа, якому буде привласнюватись ***результат операції***, має бути достатньою для зберігання результату, і об'явлена заздалегідь. Довжина хоча б одного з ***операндів*** повинна дорівнювати довжині результату (явно розширена до довжини результату). Інший операнд може бути автоматично розширено до цієї довжини. При додаванні 2-х n-розрядних чисел довжина результату має бути n+1. При множенні 2-х n-розрядних чисел довжина результату має бути 2* n.

Для збільшення розрядності операндів до необхідної довжини існує функція ***resize***. Функція ***resize (<число>,<кількість розрядів>)*** розширює число до кількості розрядів, заданої цілим десятковим числом.

В якості числа може бути ім'я змінної, сигналу або константи з зазначенням її типу. При цьому число типу unsigned доповнюється зліва відповідною кількістю нулів, а типу signed – значенням його знакового (старшого) розряду.

Наприклад:

resize(unsigned (“10”), 4) =”0010”, resize(signed (“10”), 4) =”1110”.

resize(unsigned (“01”), 4) =”0001”, resize(signed (“01”), 4) =”0001”.

При описі пристрою рекомендується сигнали в його ***інтерфейсі*** оголошувати з типом std_logic або bit, а в архітектурі використовувати ***внутрішні*** сигнали або змінні з типом unsigned і signed. Для такого перетворення використовуються функції *конкретизації типів*, так як ці типи мають однакове векторне представлення. Для перетворень типів між

Integer та Unsigned і Signed *є функції перетворення типів*: To_Unsigned, To_Signed, To_Integer, так як ці типи мають різне представлення.

Наведемо ряд прикладів:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.numeric_std.all;

entity lab5 is
generic (Lin: integer :=2);
port(
A,B: in STD_LOGIC_vector (Lin downto 0);
C: in STD_LOGIC;
Sum: out STD_LOGIC_vector (Lin+1 downto 0));
end lab5;

architecture lab5 of lab5 is
signal D: signed (Lin+1 downto 0);      -- 4 розряди
signal DU: unsigned(Lin downto 0):=(others=>'0');
signal As, Bs: signed (Lin downto 0);       -- 3 розряди
signal Mul: signed (Lin*2+1 downto 0);        -- 6 розрядів
signal Int, Intsum: integer;
begin
D<=resize(As, Lin+2)+ Bs after 2ns;
Mul<=As*3;
process (C)
          variable vsum:unsigned (Lin+1 downto 0);
          variable va,vb:unsigned (Lin downto 0);
begin
     if Rising_Edge(C) then
          DU<=DU-1 after 2ns; -- працює як звичайний лічильник (11..1-> 00..0)
          va:=unsigned(A); vb:=unsigned(B); --конкретизація типів
          vsum:=resize(va,Lin+2) - resize(vb,Lin+2); -- розширення операндів до довжини
                                                        -- результату
          Sum<=STD_LOGIC_vector(vsum) after 2ns; --конкретизація типів
               else NULL;
               end if;
end process;
----------перевірка----------
```

```
int<=to_integer(signed(D));          --перетворення типів
intsum<=to_integer(signed(Sum));
end lab5;
```

Якщо арифметичних операцій в моделі небагато, можна одразу використати подвійне перетворення типів даних, не вводячи проміжні змінні. Наприклад,

```
signal Ain, Bout: STD_LOGIC_vector (Lin downto 0);
Bout <= std_logic_vector(unsigned(Ain) + 1);
```

Наприкінці, кілька слів про більш ранні пакети арифметичних операцій, які передували пакетам Numeric_std та Numeric_bit. Це пакети Std_Logic_arith, Std_Logic_signed, Std_Logic_unsigned із бібліотеки IEEE.

У пакеті std_logic_arith визначено типи unsigned та signed як вектори елементів типу std_logic та підтип small_int типу integer, а також ряд арифметичних функцій, функцій порівняння та зсуву над операндами цих типів. Пакети Std_Logic_signed і Std_Logic_unsigned в основному включають такі ж функції, як і Std_Logic_arith, але вони призначені для опису таких об'єктів проекту, в яких використовуються або тільки тип signed, або тільки тип unsigned. При використанні всіх цих пакетів в операціях для двох аргументів в якості другого операнду може бути Std_Logic_vector.