Міністерство освіти і науки України
Вінницький національний технічний університет

Й. Й. Білинський, Б. П. Книш

ЦИФРОВА СХЕМОТЕХНІКА
Електронно-обчислювальні пристрої

Навчальний посібник

Вінниця
ВНТУ
2021

УДК [004.31+621.38](075.8)
Б61

Рекомендовано до друку Вченою радою Вінницького національного технічного університету Міністерства освіти і науки України (протокол № 15 від 31.05.2021 р.)

Рецензенти:

В. М. Кичак, доктор технічних наук, професор

В. М. Смолій, доктор технічних наук, професор

В. В. Мартинюк, доктор технічних наук, професор

Білинський, Й. Й.

Б61 Цифрова схемотехніка. Електронно-обчислювальні пристрої : навчальний посібник / Й. Й. Білинський, Б. П. Книш. – Вінниця : ВНТУ, 2021. – 66 с.

ISBN 978-966-641-865-7

У навчальному посібнику наведено загальні принципи побудови арифметико-логічних пристроїв, операційних пристроїв, пристроїв керування, програмованих логічних матриць та інтегральних схем, розглянуто їх особливості та функціональні можливості.

Метою навчального посібника є ознайомлення читача з сучасними електронно-обчислювальними пристроями.

Посібник розроблено відповідно до навчальної програми з дисциплін «Цифрова схемотехніка» та «Схемотехніка» й розраховано для студентів спеціальностей 153 – «Мікро- та наносистемна техніка» та 171 – «Електроніка».

УДК [004.31+621.38](075.8)

ISBN 978-966-641-865-7

ЗМІСТ

СПИСОК АБРЕВІАТУР

АЛБ – арифметико-логічний блок
АЛП – арифметико-логічний пристрій
АНМК – адреса наступної мікрокоманди
БЛЕ – блок логічних елементів
БМК – базовий матричний кристал
БО – базовий осередок
БОЗ – блок обробки знаків
БОМ – блок обробки мантис
БОП – блок обробки порядків
ДШМКО – дешифратора мікрооперацій
КА – керувальний автомат
КЛБ – конфігурований логічний блок
КМ – комутаційна матриця
КМКО – код мікрооперації
КМОН – комплементарний метал-оксид-напівпровідник
КОП – код операції
КС – комбінаційна схема
КУ – код умови.
ЛТ – логічна таблиця
ЛчЦ – лічильник циклу
МБО – матричний базовий осередок
МКПЛ – мікропрограмний лічильник
МЛБ – матричний логічний блок
МПА – мікропрограмний апарат
НВІС – надвелика інтегральна схема
ОА – операційний автомат
ОБ – операційний блок
ОЗП – оперативний запам’ятовувальний пристрій
ОМ – обчислювальна машина
ОП – основна пам’ять
ПБО – периферійний базовий осередок
ПЗП – постійний запам’ятовувальний пристрій

ПК – плаваюча кома
ПЛІС – програмована логічна інтегральна схема
ПЛМ – програмована логічна матриця
ПМЛ – програмована матрична логіка.
РгК – регістр команди
РгМК – регістр мікрокоманди
РДк – регістр дільника
РДн – регістр діленого
РЗП – регістр загального призначення
РМк – регістр множника
РМн – регістр множеного
РСЧД – регістр суми часткових добутків
РЧ – регістр частки
САПР – система автоматизованого проектування
СМ-МП – суматор-мультиплексор
СПЛП – складний програмований логічний пристрій
СФА – схема формування адреси
СЧД – сума часткових добутків
ТЗ – технічне завдання
УФ – ультрафіолет
ФК – фіксована кома
ФО – функціональний осередок
ФО – функціональний оператор
ЦПК – центральний пристрій керування

ВСТУП

Більшість сучасних систем автоматики, обчислення, передачі та обробки інформації виконується на пристроях цифрової техніки. Тому знання про принципи використання цифрових пристроїв і побудови на їх основі систем різноманітного призначення є актуальним і мають велику практичну цінність, причому, як в інженерній діяльності, так і при методологічних дослідженнях.

Цифрова схемотехніка – це фундаментальна галузь науки, техніки й виробництва, що охоплює електронні пристрої і системи, компоненти сучасних комп'ютерних систем та електронно-обчислювальні пристрої.

Електронно-обчислювальний пристрій у вузькому значенні – це електронний пристрій з можливістю програмування (або, за старою назвою, – «електронна обчислювальна машина»), що проводить обчислення за попередньо визначеним алгоритмом.

До електронно-обчислювальних пристроїв відносять арифметико-логічні пристрої, операційні пристрої (однотактові, багатотактові, конвеєрні, таблично-алгоритмічні), пристрої керування (центральний, з жорсткою логікою, мікропрограмного керування), програмовані логічні матриці та пристрої на їх основі, програмовані логічні інтегральні схеми.

Тому метою навчального посібника є ознайомлення читача з сучасними електронно-обчислювальними пристроями.

Посібник базується на курсі лекцій з дисциплін «Цифрова схемотехніка» та «Схемотехніка».

Посібник складається з п'яти розділів, що містять загальні відомості про арифметико-логічні пристрої, операції арифметико-логічних пристроїв, операційні пристрої, пристрої керування, програмовані логічні матриці та програмовані логічні інтегральні схеми.

Посібник адресований широкому колу читачів, які займаються розробкою електронно-обчислювальних пристроїв, і розрахований на студентів спеціальностей 153 – «Мікро- та наносистемна техніка» та 171 – «Електроніка»; може бути корисним студентам інших спеціальностей.

1 ЗАГАЛЬНІ ВІДОМОСТІ ПРО АРИФМЕТИКО-ЛОГІЧНІ ПРИСТРОЇ

1.1 Функції арифметико-логічних пристроїв

Арифметико-логічний пристрій (АЛП) виконує арифметичні, логічні й інші операції обробки даних над операндами, які являють собою двійкові числа з фіксованою та рухомою комами, двійково-десяткові числа, команди, адреси, алфавітно-цифрові коди.

АЛП працює на основі мікропрограмного керування. Кожна машинна операція поділяється на послідовність елементарних дій (передавання слів, інверсія слів тощо), реалізованих як такти.

Функціональні елементарні обчислення, які виконуються в одному машинному такті, називаються мікрооперaціями.

Мікрооперація визначається відповідним сигналом керування. Комплекс мікрооперацій, які виконуються в одному такті, є мікрокомандою. Крім того, мікрокоманда може містити або одну мікрооперацію, або жодної.

Вибір порядку виконання мікрооперацій відбувається шляхом аналізу логічних умов, які можуть набувати значення «1» (так) або «0» (ні) залежно від значень операндів і результатів обчислень. Мікроалгоритм операції, що записаний в термінах мікрооперацій та логічних умов, є мікропрограмою. Кожній машинній операції властива своя мікропрограма.

Сукупністю операційного та керувального пристроїв може бути будь-який цифровий обчислювач, зокрема й АЛП.

Операційний пристрій забезпечує виконання арифметико-логічних операцій. В пристрої керування виконуються операції за допомогою послідовних сигналів керування, що генеруються ним залежно від мікропрограми. В математичних моделях АЛП перший пристрій подається операційним автоматом (ОА), а другий – керувальним автоматом (КА) (рис. 1.1).

ОА приймає на вхід A операнди, на вхід Y – сигнали керування $\{Y_i\}$, а потім передає на вихід Z результати операції та формує множину значень логічних умов $\{X_i\}$.

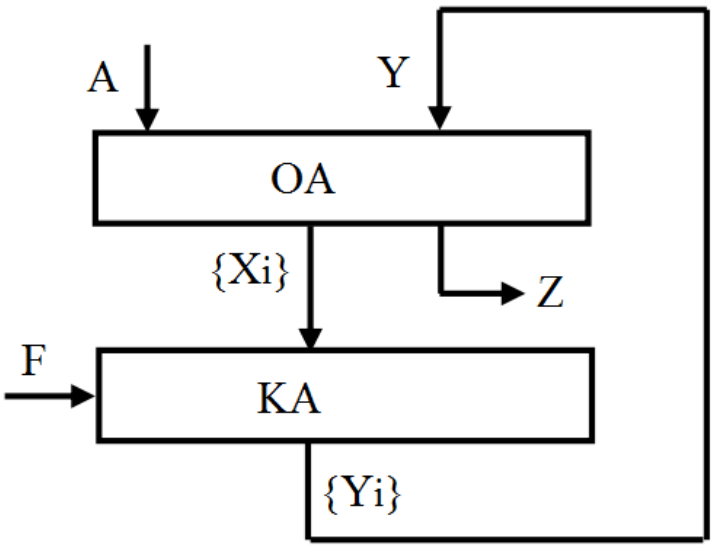


Рисунок 1.1 – Структура математичної моделі АЛП

КА приймає на вхід X логічні умови $\{X_i\}$ та, залежно від їх значень та коду операції, на вході F формується послідовність сигналів керування $\{Y_i\}$.

1.2 Класифікація арифметико-логічних пристроїв

АЛП класифікують за такими ознаками:

- способом обробки даних – паралельні, послідовні, паралельно-послідовні;
- системою числення – 2-кові, 8-кові, 10-кові, а також пристрої на основі спеціальних систем (залишкових класів, зі штучним порядком ваги, чисел Фібоначчі) тощо;
- формою подання чисел – з плаваючою комою (ПК), з фіксованою комою, цілі двійкові і десяткові числа;
- часом виконання операцій – синхронні та асинхронні;
- способом виконання мікрооперацій – із закріпленими мікрооперація-ми, зі спільними мікрооперація-ми;
- типом керувального автомата – зі схемною чи програмованою логікою;
- методом побудови – багатофункціональні та блокові.

В асинхронних АЛП час виконання залежить від типу операції, а в синхронних – на виконання різних операцій надається єдиний інтервал часу.

В АЛП із закріпленими мікрооперація-ми кожен регістр за допомогою додаткових комбінаційних схем (КС) виконує певні мікрооперації. При цьому КС досить часто повторюються, що, в свою чергу, вимагає значних апаратних витрат.

В АЛП із спільними мікрооперація-ми можна виділити запам’ятовувальний блок, а саме: блок регістрів, в якому здійснюються однорідні мікрооперації (прийом операндів, їхнє зберігання та передача), і

комбінаційний блок, в якому сконцентровані всі схеми виконання мікрооперацій (формування кодів, зсуви, додавання тощо). Обидва блоки взаємодіють між собою за допомогою мультиплексорів і демультиплексорів. АЛП із спільними мікрооперaціями ще називають магістральними (до числових магістралей підключаються регістри по черзі).

Багатофункціональні (універсальні) АЛП виконують весь список операцій, що досягається відповідним налаштуванням та комутацією вузлів. Блокові АЛП формуються з окремих блоків, які орієнтовані на виконання різних типів операцій (блок множення чисел з ПК). Вони можуть використовуватись у комп'ютерах з високою продуктивністю.

В АЛП можуть використовуватись два типи КА:

- зі схемною («жорсткою») логікою, що складається з елементів пам'яті (тригерів) і КС, які генерують відповідні елементам сигнали керування $\{Y_i\}$;
- з програмованою логікою, при якій для кожної операції в спеціальній пам'яті записується мікропрограма у вигляді набору мікрокоманд (послідовності керувальних слів), який містить дані про мікрооперації, що мають здійснюватись в даному такті, та адресу наступної мікрокоманди.

1.3 Структури арифметико-логічних пристроїв

Найпоширенішу та узагальнену структуру АЛП для виконання арифметичних операцій наведено на рис. 1.2.

ОА універсальних комп'ютерів складаються з:

- арифметико-логічного блоку (АЛБ);
- набору регістрів загального призначення (РЗП);
- блоку контролю.

В АЛБ можна виділити комбінаційний суматор SM, вхідні регістри А і В для прийому операндів і вихідний регістр З для запису результату. В АЛБ є логічні схеми, що генерують множини $\{X_i\}$ сигналів логічних умов.

Регістри загального призначення використовуються для прийому та зберігання операндів, а також проміжних і кінцевих результатів.

Блок контролю забезпечує перевірку правильності виконання арифметико-логічних операцій при одночасній реалізації тієї ж самої команди за допомогою дублювальної апаратури. Також блок контролю забезпечує порівняння результатів шляхом здійснення дій над спеціальними кодами, які отримані від операндів при додаванні за модулем 2, 3 тощо.

У випадку виявлення помилок чи збоїв у роботі ОА блок контролю надсилає до КА код помилок $\{K_i\}$.

На АЛП надходить код операції від центрального пристрою керування. Пристрої керування зі схемною логікою в АЛП прискорюють виконання операцій. КА з програмованою логікою забезпечує гнучкість мікропрограмування та дозволяє змінювати структуру мікропрограми при введенні нових команд. На сьогодні в конструкціях АЛП можуть поєднуватись обидва типи КА.

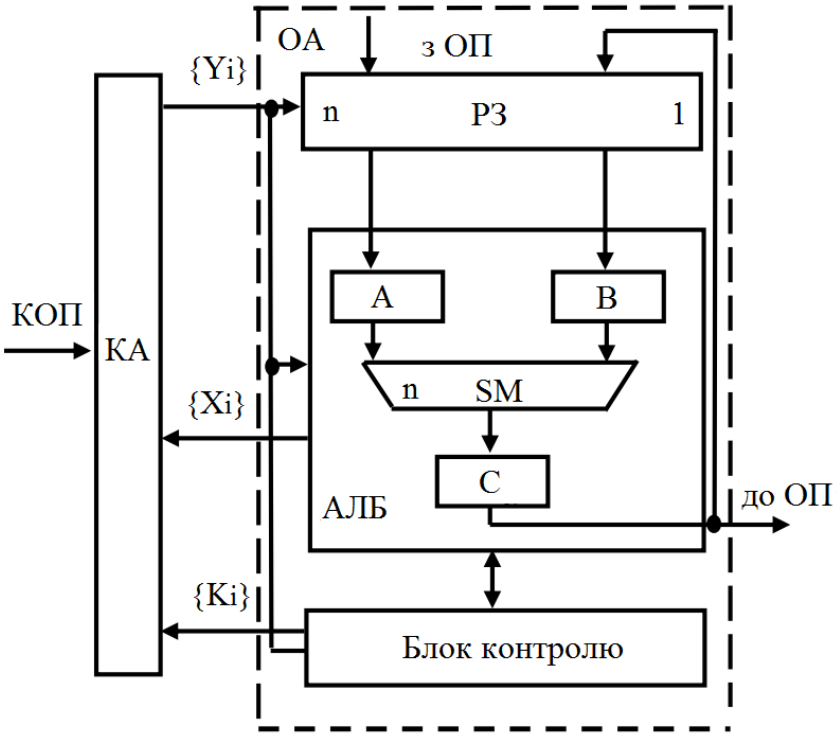


Рисунок 1.2 – Узагальнена структура АЛП
для виконання арифметичних операцій

Узагальнену структуру АЛП для виконання логічних операцій наведено на рис. 1.3.

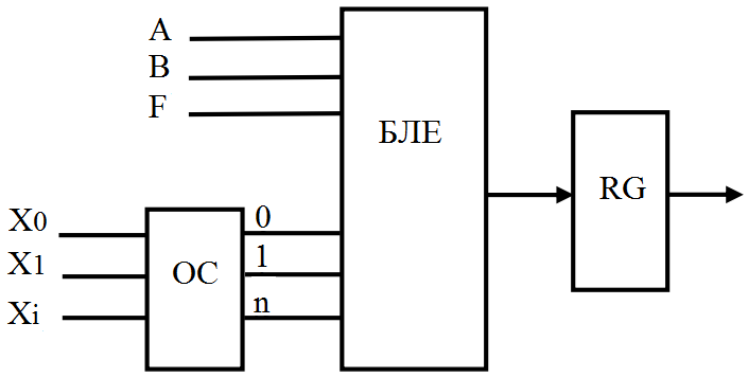


Рисунок 1.3 – Узагальнена структура АЛП
для виконання логічних операцій

До складу ОА в універсальних комп’ютерах входять:

- блок логічних елементів (БЛЕ), який забезпечує виконання логічних операцій І, АБО, виключне АБО тощо;
- дешифратор команд (DK);
- операнди (A, B, F);
- коди операції (X_0, X_1, X_2, X_i);
- вихідний регістр (RG).

Приклад простої реалізації однорозрядного АЛП на логічних елементах для реалізації операцій кон'юнкції (A^B), диз'юнкції (AVB) та виключного АБО (A⊕B) наведено на рис. 1.4.

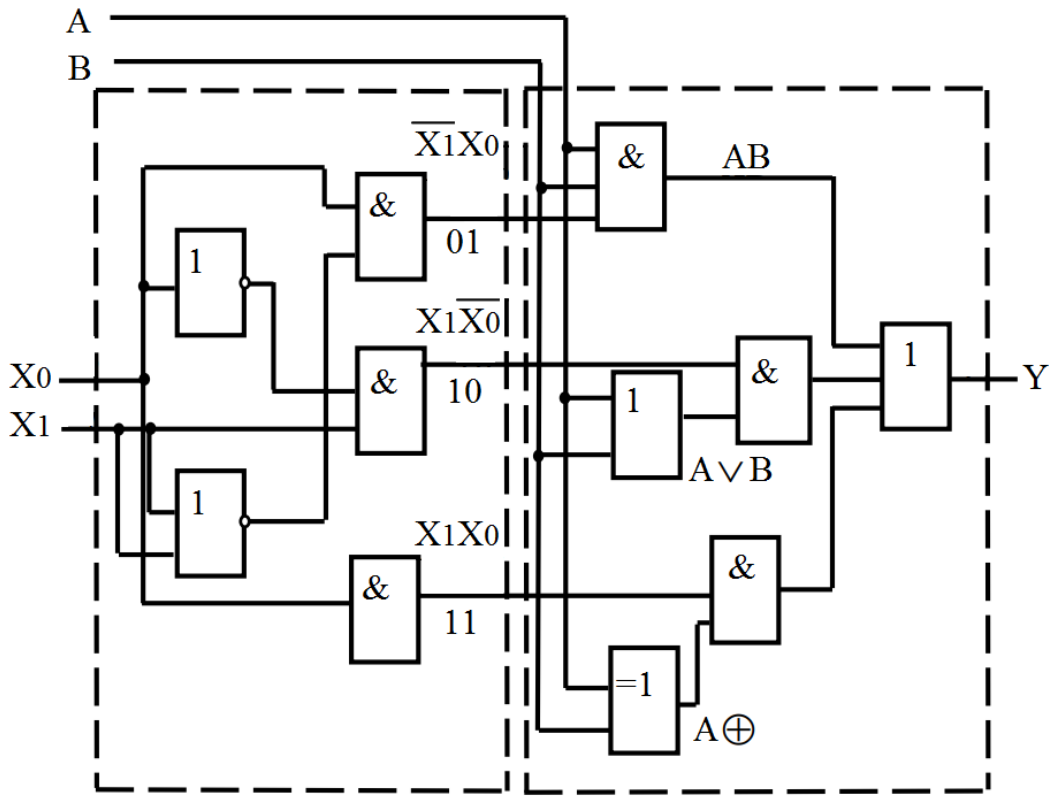


Рисунок 1.4 – Структурна схема реалізації однорозрядного АЛП на логічних елементах для реалізації логічних операцій кон'юнкції, диз'юнкції та виключного АБО

Приклад реалізації математичних операцій додавання та віднімання дворозрядних двійкових чисел наведено у вигляді структурної схеми (рис. 1.5).

Вищенаведена схема містить в собі:

1. 3-розрядний двійковий лічильник;
2. Дешифратор на 3 входи;
3. Регістр операнда A;
4. Регістр операнда B;
5. Перетворювач прямого кода в обернений;
6. Ключі на логічних елементах I;
7. Суматор першого розряду;
8. Суматор другого розряду;
9. Регістр суми;
10. Ключі на логічних елементах I.

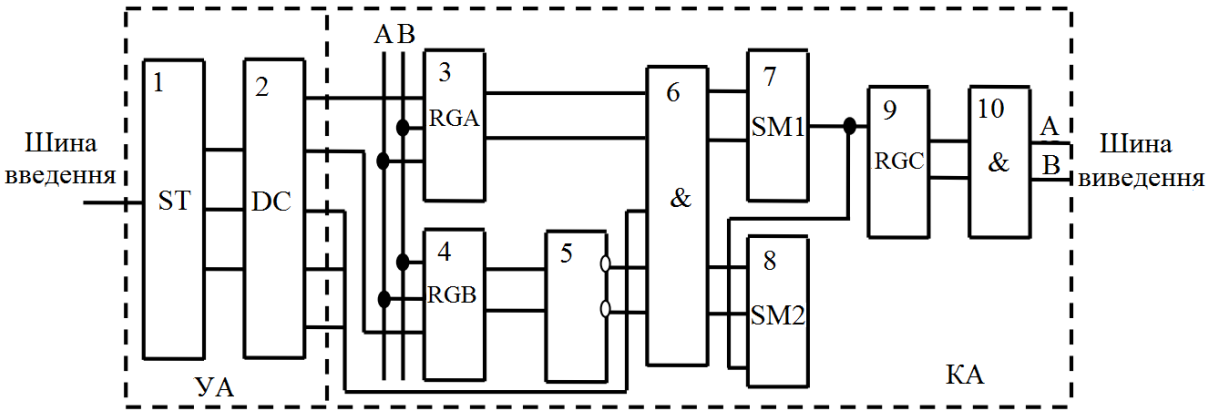


Рисунок 1.5 – Структурна схема реалізації математичних операцій додавання та віднімання дворозрядних двійкових чисел

КА складається з дворозрядного двійкового лічильника імпульсів, який виконаний на тригерах, та дешифратора, виконаного на логічних елементах.

Математичні операції виконуються за 5 тактів:

1. Записування операнда в регістр А;
2. Записування операнда в регістр В;
3. Подача сигналу дозволу додавання;
4. Записування операнда в регістр С;
5. Видача сигналу на вихідну шину.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що означає мікрооперація?
2. Що означає мікрокоманда?
3. Яке призначення ОА?
4. Яке призначення пристрою керування?
5. За якими ознаками класифікують АЛП?
6. Які типи пристроїв керування застосовують в АЛП?
7. Складіть схему загальної структури АЛП та опишіть призначення її блоків.

2 ОПЕРАЦІЇ АРИФМЕТИКО-ЛОГІЧНИХ ПРИСТРОЇВ

2.1 Операції арифметико-логічних пристроїв

2.1.1 Елементарні операції

Складні операції в АЛП здійснюються як послідовність елементарних операцій.

До типових елементарних операцій відносять:

1. Зсув – це зміщення кодів, що зберігаються в регістрі, вліво чи вправо на задане число розрядів;
2. Операція рахунку – це додавання до слова «1» або «-1»;
3. Шифрування – це перетворення однорядного коду в двійковий код;
4. Дешифрування (однорядний код) – це перетворення слів в сигнали;
5. Порівняння – це визначення старшого розряду двох слів чи їх рівності;
6. Порозрядне доповнення – це формування оберненого коду;
7. Логічне множення і додавання двох слів по розрядах;
8. Додавання двох слів за модулем по розрядах;
9. Сума двох слів.

АЛП формується на основі КС, що можуть виконувати елементарні операції. На основі КС для виконання елементарних операцій формуються вузли АЛП для виконання складних операцій.

2.1.2 Складні операції

До типових складних операції АЛП відносять:

1. Логічні операції над двійковими числами;
2. Операції зсуву (зсуву вправо чи вліво) на певну кількість розрядів, що задається;
3. Арифметичні операції над двійковими числами;
4. Операції відношення (менше, більше, менше-рівне, більше-рівне);
5. Операції обчислення елементарних функцій типу *exp(x)*, *In(x)*, *sin(x)*, *cos(x)*, *arctg(y/x)*;
6. Операції обробки символів і рядків символів.

Реалізація складних операцій в АЛП вимагає подання алгоритму виконання операції в формі, що ілюструє базові обчислювальні та структурні характеристики. Обчислювальні характеристики можна описати повним набором функціональних операторів (ФО) алгоритму та відповідними затримками, які можна обчислити. Структурні характеристики можна описати зв’язками між функціональними операторами алгоритму та їх взаємозалежністю.

2.2 Алгоритми виконання операцій арифметико-логічних пристроїв

Для сучасних комп'ютерів загальноприйнятним є формат з фіксованою комою (ФК). При ньому кома перебуває у фіксованому стані з правої сторони від молодшого розряду коду числа. З ФК можуть бути числа як без знака (всі *n* позицій числа відводяться під значущі цифри), так і зі знаком. В останньому випадку старший (*n*-1)-й розряд числа приймає знак числа (0 – плюс, 1 – мінус), а під значущі цифри відведено розряди з (*n*-2)-го по 0-й. Під час записування від'ємних чисел використовується доповняльний код.

2.2.1 Алгоритм виконання операцій додавання та віднімання кодів чисел з фіксованою комою

На рис. 2.1 показано можливу структуру пристрою додавання та віднімання кодів чисел з фіксованою комою.

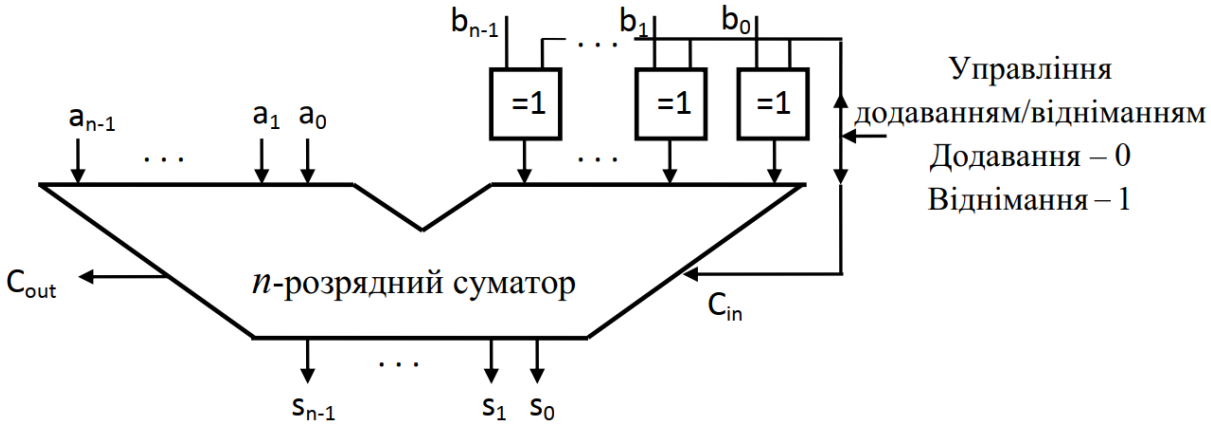


Рисунок 2.1 – Структурна схема пристрою додавання та віднімання

Центральне місце в пристрої займає *n*-розрядний двійковий суматор.

Операнд *A* надходить на вхід суматора без змін. Операнд *B* проходить заздалегідь через схеми додавання за модулем 2, тому код *B*, що надходить на інший вхід суматора, залежить від операції, яка виконується. При заданій операції додавання (код керування – 0) результат на виході визначається виразом $S = A + B$. При заданій операції віднімання (код керування – 1), на вхід суматора надходять інверсні значення всіх розрядів *B* і, також, на вхід перенесення в молодший розряд суматора C_{in} надходить 1. Таким чином, на виході буде вираз $S = A + \overline{B} + 1$, що рівнозначно додаванню до числа *A* числа *B* з протилежним знаком, тобто відніманню.

2.2.2 Алгоритм виконання операції множення кодів чисел з фіксованою комою

Алгоритм може бути реалізований за допомогою схеми, як показано на рис. 2.2.

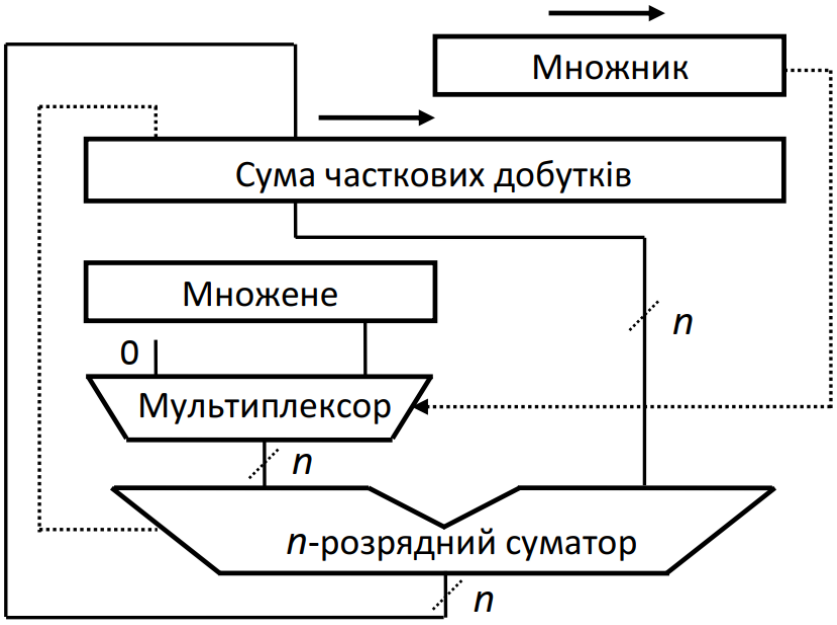


Рисунок 2.2 – Структурна схема пристрою множення

На початку множене і множник заносяться в *n*-розрядні регістри множеного (РМн) та множника (РМк), відповідно, а розряди 2*n*-розрядного регістра суми часткових добутків (РСЧД) визначаються як 0. Множення відбувається протягом *n* кроків. Під час кожного кроку, залежно від стану молодшого розряду РМк, який керує мультиплексором, на один з входів *n*-разрядного суматора надходить множене або 0. На інший вхід надходить вміст *n* старших розрядів РСЧД. Таким чином, новий частковий добуток з суматора надходить до старших розрядів РСЧД. Потім вміст РСЧД зміщується на 1 розряд вправо. При цьому в старший розряд звільненого регістра надходять значення переносу зі старшого розряду суматора. Мультиплексор керується молодшим розрядом РМк, тому вміст цього регістра також зміщується на 1 розряд вправо. Ця послідовність повторюється *n* разів.

Існує інше рішення, яке є більш економічним щодо продуктивних можливостей апаратури. За ним замість 2-х регістрів – *n*-розрядного РМк і 2*n*-розрядного РСЧД – застосовується 1 комбінований 2*n*-розрядний регістр. Таким чином, множник спочатку надходить в молодші *n* розряди цього регістра, а старші розряди занулюються. У міру зміщення вправо молодші розряди множника, що вже були проаналізовані, виходять з регістра, звільняючи місце для чергової цифри, яка визначає суму часткових добутків (СЧД). Зазвичай такий регістр формується з 2-х *n*-розрядних регістрів, які об’єдні ланцюгами зсуву. Також варто додатково

відзначити, що якщо чергова цифра множника дорівнює 1, то для знаходження СЧД вимагається здійснення операції додавання та зсуву, а при цифрі множника, що дорівнює 0, можна обійтися без додавання, обмежившись лише зсувом.

2.2.3 Алгоритми виконання операції ділення кодів чисел з фіксованою комою

Алгоритм ділення кодів чисел з фіксованою комою може бути реалізований за допомогою схеми, яка наведена на рис. 2.3.

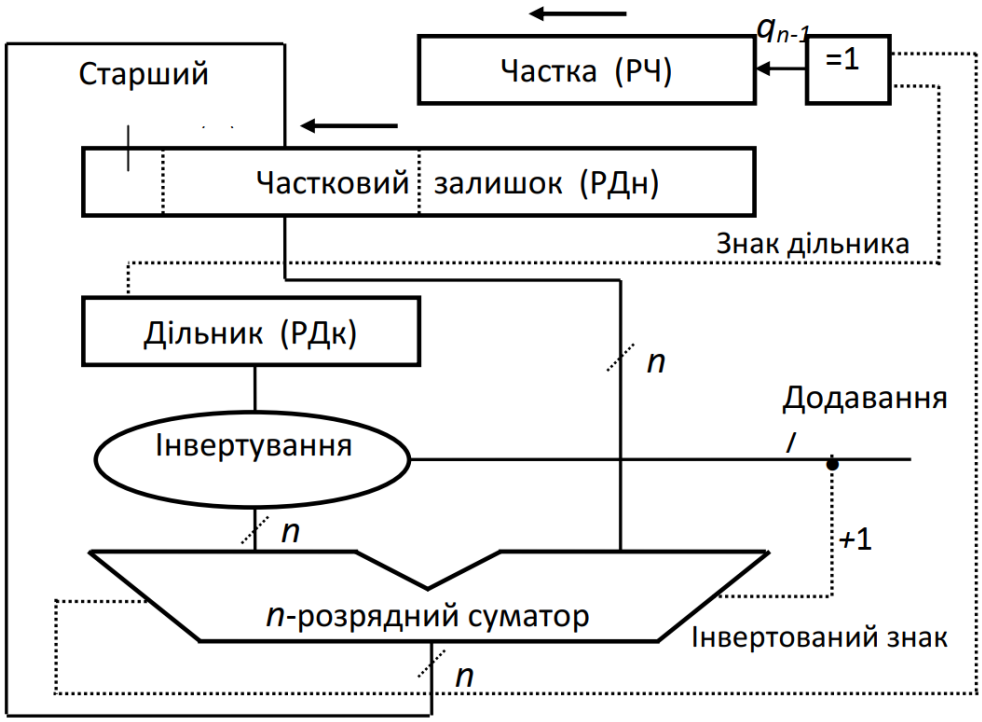


Рисунок 2.3 – Структурна схема пристрою ділення

Операція ділення починається з надходження діленого в 2*n*-розрядний регістр діленого (РДн) та дільника в *n*-розрядний регістр дільника (РДк). В лічильник циклу (ЛчЦ, на схемі не наведено), який слугує для підрахунку кількості цифр частки, що надійшли на лічильник, надходить початкове значення, яке дорівнює *n*.

На кожному етапі операції ділення вміст РДн і регістра частки (РЧ) зсувається на 1 розряд вліво. Залежно від комбінації знаків часткового залишку та дільника формується значення чергової цифри частки та потрібна дія, а саме: віднімання чи додавання дільника. Віднімання дільника відбувається з допомогою додавання доповняльного коду дільника. Перетворення в цей код відбувається за рахунок надходження дільника на вхід суматора з оберненим кодом з подальшим додаванням 1 до молодшого розряду суматора.

Ця процедура повторюється до закінчення всіх цифр діленого, на що буде вказувати нульовий вміст ЛчЦ. В кінці операції ділення частка роз-

міщується в регістрі частки, в той час як в регістрі діленого буде знаходитись залишок від ділення.

На завершальному етапі, якщо це потрібно, відбувається коригування отриманого результату.

На практиці для накопичення та зберігання частки замість окремого регістра застосовують молодші розряди регістра діленого, які вивільняються в процесі зміщень.

2.2.4 Особливості виконання операцій в арифметико-логічних пристроях з плаваючою комою

Операції над числами з ПК характеризуються істотними відмінностями від аналогічних операцій цілочислової арифметики, тому вони, зазвичай, реалізуються за допомогою самостійного операційного пристрою. Як і у випадку цілочислового ОПр, операційний пристрій для чисел з ПК мінімально має забезпечувати виконання 4-х арифметичних дій: додавання, віднімання, множення та ділення.

Затвердження стандарту IEEE 754 привело до того, що вимогою до всіх обчислювальних машин (ОМ) стало забезпечення операцій з числами, які надходять у форматах, що визначені в даному стандарті. Основні положення запису чисел згідно зі стандартом IEEE 754 полягають у тому, що мантиси чисел M подаються у нормалізованому вигляді і при цьому відбувається прийом прихованого розряду, коли старша цифра мантиси (завжди 1) в записі числа відсутня, тобто в полі мантиси старшою є друга старша цифра нормалізованої мантиси.

Загальноприйнята умова нормалізації передбачає $S = |M| < 1$, а в стандарті IEEE 754 застосовується умова $1 = |M| < 2$. Запис числа передбачає зміщений порядок (збільшений на величину зміщення, яке для стандарту IEEE 754 для одинарного формату дорівнює 127, а для подвійного – 1023).

Враховуючи ці особливості будь-яку арифметичну операцію над числами у форматі з ПК можна подати у вигляді

$$\pm Z_M \times 2^{Z_{СП}} = \left(\pm X_M \times 2^{X_{СП}}\right) \Diamond \left(\pm Y_M \times 2^{Y_{СП}}\right),$$

де X_M, Y_M, Z_M – нормалізовані мантиси операндів і-го розряду,

$X_{СП}$, $Y_{СП}$, $Z_{СП}$ – зміщені порядки операндів і-го розряду,

$\Diamond$ – знак арифметичної операції.

Незважаючи на всі ці відмінності у виконанні різних арифметичних операцій підготовчий та завершальний етапи в усіх випадках однакові. Тому є сенс розглянути їх окремо один від одного.

На підготовчому етапі першою особливістю операційних пристроїв для чисел з ПК є те, що в них операції для трьох складових чисел з ПК (знаками, мантисами та порядками операндів) виконуються окремо: блоком обробки знаків (БОЗ), блоком обробки порядків (БОП) та блоком обробки мантис (БОМ). Зберігання операндів та результату в основній пам’яті (ОП)

передбачає відповідні регістри. Незважаючи на те, що ці регістри можуть бути реалізовані фізично у вигляді єдиних пристроїв, кожен з них потрібно розглядати як сукупність 3-х регістрів: знака, порядку та мантиси. З цього випливає, що на етапі надходження операндів у регістри ОП відбувається «розпакування» чисел з ПК та розбиття їх на 3 складові. У ході цього процесу в старшому розряді регістра мантиси відновлюється 1, яка при записуванні числа була відсутня або прихована.

Також може бути виконана перевірка на рівність 0 одного чи обох операндів (згідно зі стандартом IEEE 754 подання нульового значення передбачає такий запис числа, в якому 0 дорівнюють всі розряди порядку). Це дозволяє усунути непотрібні операції. Наприклад, в операціях множення та ділення, якщо 0 дорівнюють множник, множене чи ділене, як результат можна прийняти нульове значення відразу.

Завершальний етап передбачає, що виконання будь-якої арифметичної операції зводяться до виявлення нульового значення мантиси, її нормалізації, виявлення від'ємного переповнювання порядку, «упакування» складових результату.

Нульове значення мантиси виникає, наприклад, під час додавання чи віднімання мантис. Іншою причиною може бути зміщення мантиси вправо для усунення переповнення. В обох випадках відбувається ситуація втрати значимості мантиси, тому результатом операції стає 0. Згідно зі стандартом IEEE 754 це означає, що всі цифри порядку результату потрібно обнулити, а нормалізацію мантиси результату не потрібно проводити.

Нормалізація мантиси результату зводиться до її зміщення послідовно вліво до тих пір, поки старшу позицію не займе 1. Кожне зміщення супроводжується зменшенням на 1 порядку результату. У ході зменшення порядок може стати від'ємним. При такій ситуації результат стає рівним 0 і формується одночасно ознака втрати значимості порядку.

При завершенні мантиса результату округляється і, якщо це передбачено форматом ПК, з неї прихований розряд видаляється.

На останній фазі здійснюється «упакування» всіх складових результату (знака, порядку та мантиси), після чого такий результат надходить до вихідного регістру ОП.

2.2.4.1 Додавання та віднімання

В арифметиці додавання та віднімання з ПК – складніші операції, ніж множення та ділення. Це пов'язано з потребою вирівнювати порядки операндів. Алгоритм додавання та віднімання передбачає такі основні фази.

1. Підготовчий етап.

2. Визначення операнда, що має менший порядок, і зсув його мантиси вправо на число розрядів, яке дорівнює різниці порядків операндів.

3. Прирівнювання порядку результату до більшого з порядків операндів.

4. Додавання чи віднімання мантис та визначення знака результату.

5. Перевірку на переповнення.

6. Завершальний етап.

Додавання та віднімання виконуються однаково, але у випадку віднімання потрібно поміняти знак другого операнда на протилежний.

2.2.4.2 Множення

На початку множення чисел з ПК відбувається перевірка на рівність 0 одного зі співмножників. Якщо один з операндів дорівнює 0, то як результат видається 0, який подається у цьому форматі чисел з ПК. Наступний етап – додавання порядків. Результатом цього може стати або переповнення порядку, або втрата значимості. В обох випадках виконання операції множення припиняється та «видається» відповідне повідомлення про виникнення переривання.

У випадку, якщо порядок результату знаходиться в допустимих межах, то на наступному етапі відбувається перемножування мантис з урахуванням їх знака. При цьому це відбувається так само, як для чисел з ФК. При розташуванні добутку мантис у розрядній сітці потрібно врахувати, що мантиси подаються не цілими числами, а правильними дробами. Незважаючи на те, що результат множення мантис має подвоєну довжину (порівняно з операндами), він округляється до довжини поля мантиси.

На останньому етапі проводиться нормалізація та формування результату аналогічно процесу додавання та віднімання.

2.2.4.3 Ділення

Під час операції ділення спочатку також проводиться перевірка на 0. Якщо йому рівний дільник, то, залежно від реалізації, видається повідомлення про ділення на 0 або результатом приймається нескінченність. Якщо нулю дорівнює ділене, то результат також приймається рівним 0.

Потім виконується віднімання порядку дільника від порядку діленого, що призводить до видалення зміщення з порядку результату. Таким чином, для отримання зсунутого порядку результату до різниці потрібно додати зміщення. Після проходження цих етапів потрібна перевірка на переповнювання порядків і втрату значимості.

Наступний крок – ділення мантис. За ним йдуть нормалізація, округлення та компонування числа з мантиси й порядку.

2.3 Основні методи контролю арифметико-логічних пристроїв

При роботі АЛП характерними є відмови та збої, які здатні призводити до помилок. Під помилкою мається на увазі таке відхилення від нормальної працездатності, для відновлення якого вимагаються деякі дії з ремонту, заміни та регулювання елемента, вузла або пристрою, які є несправними. Збій – це короткочасне відхилення від нормального функціонування системи завдяки короткочасному впливу зовнішніх завад або змін параметрів елементів, яке зникає саме по собі. Надійність АЛП характеризується без-

відмовністю, достовірністю при функціонуванні та ремонтоздатністю. До головних параметрів надійності потрібно віднести середній час напрацювання на 1 відмову, середній час позбування несправності, середній час напрацювання на 1 збій, середній час відновлення правильності інформації після збою.

Методи контролю АЛП мають на меті виявляти та виправляти помилки під час їхньої роботи. Вони дають можливість усунути вплив помилок, що були виявлені, та забезпечують правильні результати на виході АЛП. Всім методам контролю характерна надлишковість, яка буває інформаційною, апаратною, часовою. Методи контролю АЛП можна умовно поділити на 2 групи: тестові й апаратні.

Для перевірки АЛП використовують спеціальні тести, тобто випробувальні програми, за результатами роботи яких оцінюють працездатність системи на період контролю. До тестів АЛП висуваються такі вимоги:

- контроль максимальної кількості вузлів і схем АЛП при найбільш завантажених режимах;
- мінімальна кількість команд в тесті;
- повторення тесту циклічно;
- фіксація команд, при роботі яких з’являються помилки.

Просте компонування тесту для контролю АЛП може бути таким. Обирається лімітована кількість операндів, над якими виконуються арифметичні та логічні операції послідовно заради отримання правильних результатів. Ці операнди та правильні результати визначаються як еталони. Далі за допомогою спеціальної програми над парами відповідних обраних операндів виконуються послідовно всі операції, що їх реалізує АЛП, з порівнянням отриманих результатів від кожної операції з еталоном. Зазвичай такі тести відображають лише частину неполадок і є основою для режиму профілактики АЛП. Видом програмного контролю є програмно-логічний контроль, що використовує подвійний та потрійний способи прорахунку робочих задач з одночасним порівнянням результатів, що були отримані.

Контроль передачі інформації забезпечують методи інформаційної надлишковості, які є найбільш розповсюдженими та використовують коди з виявленням і виправленням помилок. Код перевірки парності формується, коли до сукупності інформаційних розрядів додається ще 1 надлишковий розряд. Формування коду слова в контрольний розряд відбувається шляхом запису 0 або 1 так, щоб кількість 1, з урахуванням контрольного розряду, була парною (при контролі на парність) або непарною (при контролі на непарність). Далі при передачі слово передається одночасно з контрольним розрядом. Якщо пристрій прийому виявляє, що в слові, яке надійшло, не задовольняється умова парності, це ідентифікується як сигнал, що вказує на помилку. Код з перевіркою на парність дає можливість ідентифікувати всі поодинокі помилки в одному розряді, а також усі випадки непарної кількості помилок.

Одним з найпоширеніших кодів для виправлення помилок є код Геммінга. При формуванні такого коду до інформаційних розрядів слова додається деяка кількість контрольних розрядів. Під час зчитування слова апаратура контролю формує з інформаційних і контрольних розрядів число коригування, яке дорівнює 0 в разі відсутності помилки або уточнює місце помилки. Зміна стану помилкового розряду автоматично коригується на протилежний.

Контроль за виконанням арифметичних операцій здійснюється за допомогою контрольних кодів, що фактично є остачами від ділення чисел на певний модуль R (контроль за модулем R). При контролі за модулем використовуються однакові значення остачі за модулем R від результату виконання операції над операндами. Для двійкових чисел контроль за модулем можливий при $R \geq 3$. Часто використовується контроль за модулем 5. За ним можна виявляти будь-які одно-, дво-, три- та чотирирозрядні помилки і виправляти одно-, дво- та трирозрядні помилки. У разі застосування контролю за модулем більше 5 збільшується кількість кратних помилок, які можна виявити системою контролю. Крім того, потрібно враховувати, що при цьому також ростуть й апаратні витрати.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Поясніть суть виконання операції додавання кодів чисел з ПК.
2. Поясніть суть виконання операції множення кодів чисел з фіксованою комою.
3. Поясніть особливість виконання операції множення кодів чисел з ПК.
4. Поясніть суть прискореного виконання операції множення кодів чисел.
5. Поясніть суть виконання операції ділення кодів чисел з відновленням залишку.
6. Поясніть суть виконання операції ділення кодів чисел без відновлення залишку.
7. Поясніть особливість виконання операції ділення кодів чисел з ПК.
8. Як визначається знак добутку і частки кодів чисел під час виконання операцій множення та ділення кодів чисел, відповідно?
9. Які існують методи контролю АЛП?
10. Як відбувається контроль з перевіркою парності?

3 ОПЕРАЦІЙНІ ПРИСТРОЇ

3.1 Табличні операційні пристрої

Табличні операційні пристрої – це операційні пристрої, в яких пошук значення результату відбувається за адресою, яка відповідає значенню операнда, в попередньо сформованій таблиці, роль якої виконує пам'ять за адресою, що дорівнює значенню операнда.

Розглянемо основний принцип формування таблиці на базі виконання операції $Y = X^3$. В таблиці 3.1 наведено двійкові коди значень операндів X, що також є адресами, та значень операндів Y, що є вмістом відповідних їм X комірок пам'яті.

Таблиця 3.1 – Значення обрахунку операції $Y = X^3$

Двійкове значення X (адреса)	Двійкове значення Y (вміст комірок пам'яті)
000	0000000
001	0000001
010	0001000
011	0011011
100	1000000
101	1111101

Розрядності X та Y обрані рівними, відповідно, 3-м та 7-ми бітам.

Запис у пам'ять потрібної інформації здійснюється розробником цифрового пристрою при використанні оперативних і перепрограмованих постійних запам'ятовувальних пристроїв (ОЗП і ПЗП, відповідно) та виробником замовних ПЗП. Важливо врахувати те, що виготовлення замовних ПЗП не вимагає від виробника таких великих затрат, з якими, зазвичай, пов'язана розробка і виготовлення спеціалізованих надвеликих інтегральних схем (НВІС). Використання перепрограмованих ПЗП, при яких допускається електричний перезапис збереженої інформації, дає можливість зміни функцій пристрою без зміни його структури.

Структуру табличного операційного пристрою наведено на рис. 3.1.

Як видно з рис. 3.1, пристрій містить вхідний n-розрядний регістр PrX, де n – розрядність вхідних даних, та вихідний m-розрядний регістр PrY, де m – розрядність вихідних даних, а також ПЗП, який зберігає табличні значення функції $Y = F(X)$.

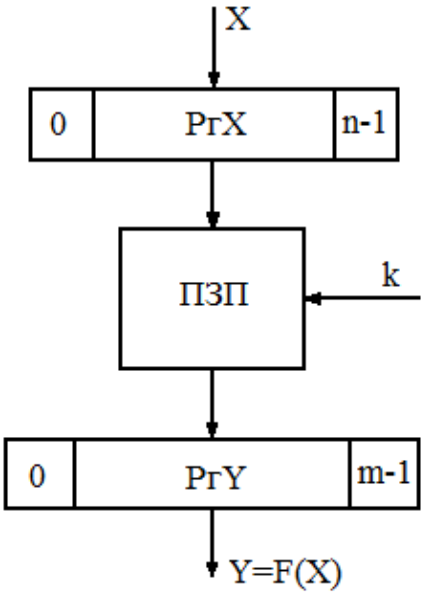


Рисунок 3.1 – Структура табличного
операційного пристрою

За потреби обчислення К функцій, таблицю кожної з них записують до ПЗП. Тоді на вхід ПЗП потрібно надіслати k-розрядний код ($k = \log_2 K$), щоб задати тип обчислюваної функції.

Табличний метод забезпечує мінімально можливий час обчислення, який зумовлений часом вибирання з пристрою пам'яті $T = t_{ПЗП}$.

Але табличний метод має недоліки, що обмежують його застосування. Основним недоліком є великий ємність пам'яті, який можна визначити з виразу $Q = K_m 2^n$. Якщо аргументи малої розрядності, то ємність ПЗП є невеликою, однак у разі обробки аргументів з великою розрядністю n і при великій кількості К обчислюваних функцій використання табличного методу є проблемним або, навіть, нереальним.

Наприклад, нехай розрядність вхідних і вихідних даних $n = m = 8$ бітів, а кількість виконуваних функцій $K = 4$. Тоді ємність ПЗП буде $Q = 4 \times 8 \times 2^8 = 1$ КВ, що цілком достатньо для реалізації. Якщо розрядність вхідних і вихідних даних $n = m = 16$ бітів, а кількість виконуваних функцій $K = 4$, тоді ємність ПЗП буде $Q = 4 \times 16 \times 2^{16} = 0{,}5$ МВ, що трохи проблематично, але також достатньо для реалізації.

Таким чином, зрозуміло, що при більшій розрядності аргумента використання табличного методу є недоцільним.

Якщо потрібно обчислювати функцію з більшою кількістю аргументів, то вищенаведена структура буде набувати вигляду, наведеного на рис. 3.2, де зображено структуру табличного операційного пристрою для виконання операцій над двома аргументами.

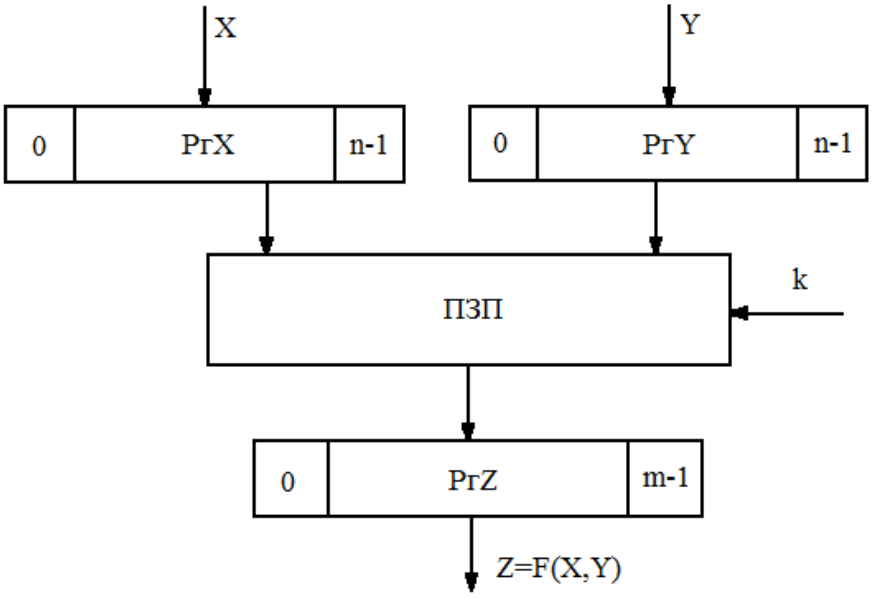


Рисунок 3.2 – Структура табличного операційного пристрою для виконання операцій над двома аргументами

Ємність пам’яті даного пристрою визначається як $Q = K_m 2^{2n}$. Для операцій з великою кількістю аргументів ємність ПЗП визначається як $Q = K_m 2^{ln}$, де l – кількість аргументів. Навіть при невеликих розрядностях аргументів ємність ПЗП є значною, тому табличний метод рідко використовується для виконання операцій над більш, ніж 1 аргументом.

Наприклад, нехай розрядність вхідних і вихідних даних n = m = 8 бітів, а кількість виконуваних функцій К = 4. Тоді ємність ПЗП буде $Q = 4 \times 8 \times 2^{16} = 256$ КВ, що цілком достатньо для реалізації. Якщо вхідних і вихідних даних n = m = 16 бітів, а кількість виконуваних функцій К = 4, тоді ємність ПЗП буде $Q = 4 \times 16 \times 2^{32} = 32$ GB, що реалізувати неможливо.

Іншими недоліками табличного методу, які обмежують його застосування, є:

- збільшення часу звертання до пам’яті при збільшенні її ємності;
- значний обсяг попередніх обчислень для розрахунку вмісту таблиць;
- значні витрати часу на запис обчислених значень у ПЗП.

Отже, застосування табличних операційних пристроїв є доцільним при малих розрядностях аргументів.

3.2 Однотактові операційні пристрої

Однотактові операційні пристрої – це операційні пристрої, в яких апаратно відображається граф виконуваного алгоритму, і задана операція реалізується шляхом одноразового проходження даних через операційні вузли. Їхню роль виконують функціональні оператори (ФО) алгоритму. Таким чином, з огляду на принципи побудови, ці пристрої можна ще назвати графо-алгоритмічними операційними пристроями.

Кожному ФО алгоритму поставлено у відповідність КС, що здійснює його виконання. На вході та виході операційного пристрою під'єднані регістри.

Часова затримка в однотактовому операційному пристрої $T_{оп}$ описується сумою затримок t_{ij} комбінаційних елементів, що знаходяться на найдовшому шляху проходження сигналу

$$T_{ОП} = \sum_{ij} MAXt_{ij} \ .$$

Цей шлях (або декілька) називається критичним шляхом. Таким чином, обчислення часу реалізації алгоритму в однотактовому операційному пристрої зводиться до знаходження критичного шляху.

Затрати обладнання W_{Qn} на однотактовий операційний пристрій описується сумою затрат обладнання W на реалізацію $КС_i$ тобто, $W_{ОП} = \sum_i W_i$, а також на вхідні та вихідні регістри.

3.3 Багатотактові операційні пристрої

Багатотактові операційні пристрої – це операційні пристрої, в яких задана операція виконується як послідовність потактового виконання ФО алгоритму, що задається мікропрограмою, яка складається з мікрокоманд для виконання елементарних операцій.

Багатотактовий операційний пристрію, структура якого наведена на рис. 3.3, складається з багатофункціональної КС, вхідних і вихідних регістрів, мультиплексорів для створення каналів передачі даних.

Багатотактовий операційний пристрій здійснює операції над 2-ма вхідними даними, тобто є двомісним, і генерує 1 вихідний результат (рис. 3.3). Спочатку операнди А та В заносяться у вхідні регістри Рг1 і Рг2 та надходять на входи багатофункціональної КС через мультиплексори МП1 та МП2, що керуються сигналами Y1 і Y2, які надходять з пристрою керування. Потім в багатофункціональній КС здійснюється задана мікрооперація, тип якої задається кодом мікрооперації, що надходить з пристрою керування. Результат мікрооперації надходить на вихід багатофункціональної КС і заноситься у вихідний регістр Рг3. З виходу цього регістра проміжний результат надходить через мультиплексори МП1 чи МП2 на 1 з входів багатофункціональної КС для здійснення наступної мікрооперації шляхом надходження з пристрою керування відповідних значень сигналів керування Y1 і Y2 на входи мультиплексорів. Запис даних до регістрів здійснюється за допомогою сигналів С1–С3 з пристрою

керування. Пристрій керування розпочинає здійснення операції після надходження на його входи сигналу про початок роботи, коду операції та тактових імпульсів ТІ. Після завершення операції на виході регістра Рг3 формується результат операції, а пристрій керування генерує сигнал завершення роботи

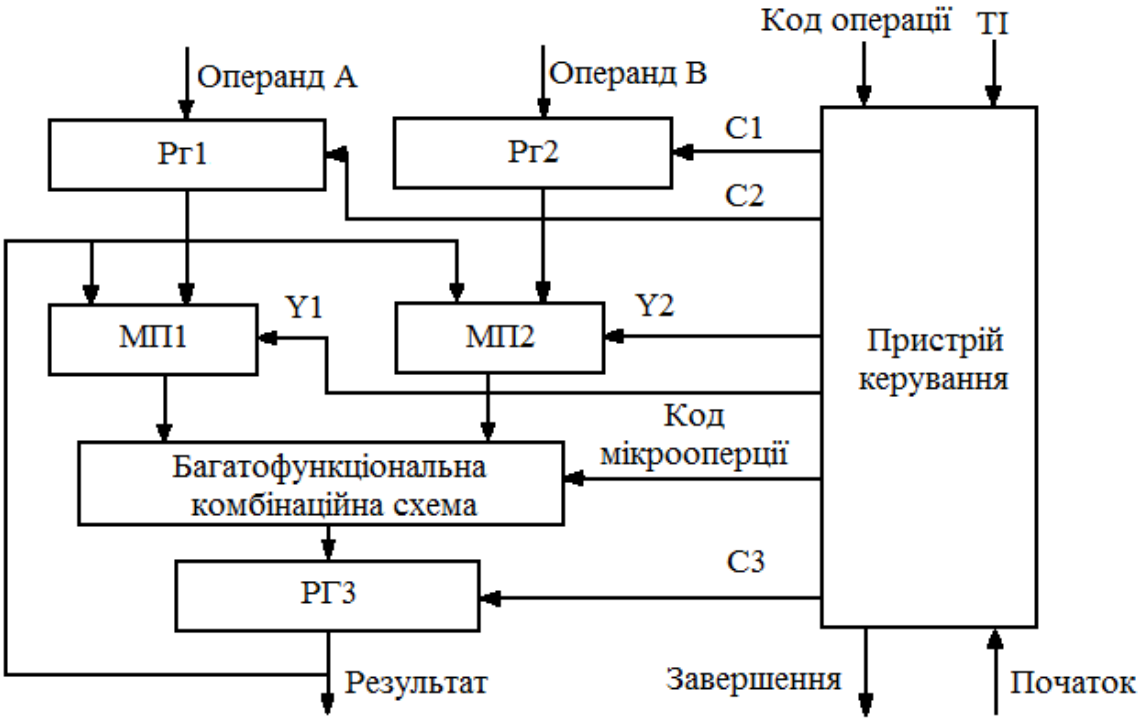


Рисунок 3.3 – Структура багатотактового операційного пристрою

Побудова багатотактового операційного пристрою приводить до виникнення 2-х задач:

- потреба в синтезі багатофункціональної КС, яка б дозволяла реалізацію всіх ФО алгоритму;
- потреба в синтезі пристрою керування, що генерує сигнали керування для виконання ФО алгоритму шляхом тимчасового запам’ятовування проміжних результатів в регістрах і направлення їх на входи багатофункціональної КС при потрібному такті.

Побудова багатотактового операційного пристрою вимагає синтезу багатофункціональної КС, яка б виконувала всі ФО. При цьому пристрій керування має визначити тип ФО, що виконується в конкретний момент часу, та забезпечити пересилання на входи багатофункціональної КС потрібних в даний момент даних.

Багатотактові операційні пристрої характеризуються малими ресурсними затратами обладнання та невисокою продуктивністю.

3.4 Конвеєрні операційні пристрої

Конвеєрні операційні пристрої – це операційні пристрої, в яких відображається апаратно потоковий граф виконуваного алгоритму з розділенням регістрами ярусів графу. Також такі пристрої можна назвати конвеєрними графо-алгоритмічними операційними пристроями.

Структуру конвеєрного операційного пристрою наведено на рис. 3.4.

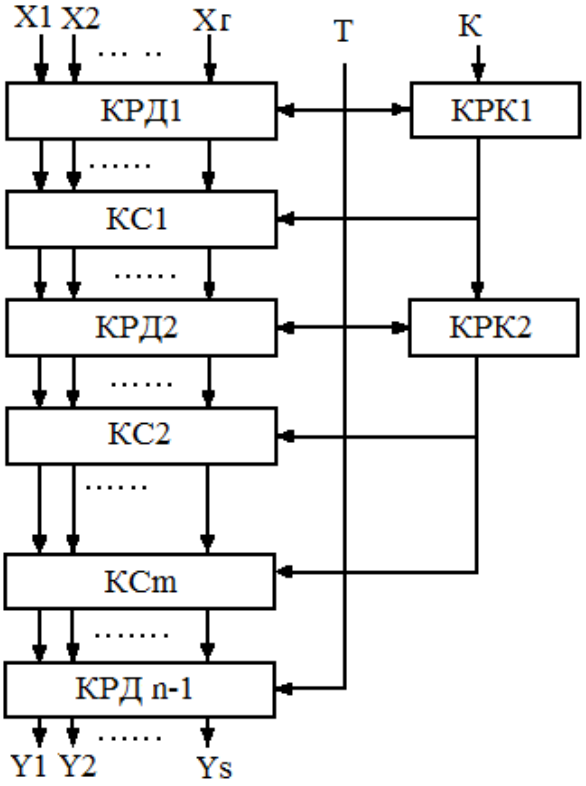


Рисунок 3.4 – Структура конвеєрного операційного пристрою

Як видно з рис. 3.4, Xi – входи даних, що надходять (i = 1, 2, ..., r), де r – кількість входів; Yj – виходи результатів обрахунку (j = 1, 2, ..., s), де s – кількість виходів; КРД і КРК – конвеєрні регістри даних Д і команд К, відповідно; m – кількість ярусів алгоритму, n – кількість ярусів конвеєра конвеєрного операційного пристрою. За умові, що структура конвеєрного операційного пристрою спрямована на реалізацію алгоритму, який обраховує лише 1 функцію, він називається однофункціональним, а групу функцій – багатофункціональним. В багатофункціональному конвеєрному операційному пристрої здійснюється програмування каналів передачі інформації та функції, які здійснюються КС відповідно до графу багатофункціонального алгоритму.

В однофункціональному конвеєрному операційному пристрої вихідні дані Y = Y1, Y2 , …, Ys визначаються однозначно вхідними даними X = X1, X2, …, Xr і функцією Ф в його структурі Y = Ф(X). В загальному

вигляді $Y = \{Y_{kf}\}$, а $X = \{X_{if}\}$, тобто, обробка здійснюється над матрицею даних. При цьому k = 1, 2, .., s; j = 1, 2, ..., r; f = 1, 2, ..., N, де r, s – кількість входів та виходів конвеєрного операційного пристрою, N – кількість груп чисел, які надходять на конвеєрний операційний пристрій. Кожному ярусу однофункціонального конвеєрного операційного пристрою відповідає A = Ф(Ai), де Ai – кількість регістрів і-го яруса, Ф – операція, яка визначається ФО (i+1)-го яруса алгоритму та виконується в (i+1)-му ярусі конвеєрного операційного пристрою. Оскільки $A_0 = X$, $A_1 = \Phi_1(A_0)$, $A_2 = \Phi_2(A_1)$ тощо, то $Y = \Phi_m(\Phi(...,(\Phi_1(X))...))$, тобто, виконання функції Ф відбувається з даними, які приходять в конвеєрний операційний пристрій через всі яруси конвеєра.

Нехай в конвеєрному операційному пристрої здійснюється обробка масиву чисел X = {X1, X2 , …, Xr}, r = 1, 2, ..., N. В першому такті за допомогою тактового імпульсу T в регістр КРД1 заноситься значення X1. Над ним в КС1 першого ярусу здійснюється операція Φ_1. Другий тактовий імпульс переписує результати цієї операції в регістр КРД2, а в регістр КРД1 заноситься значення X2. В КС1 і КС2 над значеннями, що зберігаються в регістрах КРД1 і КРД2, відповідно, здійснюються операції Φ_1 і Φ_2. Відповідно, в третьому такті результати з КС2 записуються в КРД3 і т. д. На звільнені регістри ярусів конвеєра при кожному такті записуються нові дані оброблюваного масиву. При повному завантаженні конвеєра виконуються оператори всіх m ярусів алгоритму одночасно. Стан конвеєрного операційного пристрою при і-му такті своєї роботи при обробці N чисел визначається через вектор A(t) = |A0(t)A1(t) ... Am(t)|, де t = 1, 2, ..., (m+N). Також $A_m(t) = Y_t$, тобто при кожному такті на виході конвеєрного операційного пристрою буде генеруватися результат обрахунку, але крім перших m тактів, оскільки тоді відбувається заповнення конвеєра.

Будь-який багатофункціональний конвеєрний операційний пристрій має містити певну кількість елементів, які призначені для забезпечення налаштування на задану операцію. Наприклад, КС ярусів можуть вміщувати комутатори прямих зв’язків, за допомогою яких виконується налаштування КС на виконання потрібних операцій під дією сигналів керування. Відмінність структури багатофункціонального конвеєрного операційного пристрою від однофункціонального полягає в існуванні КРК, які зберігають код операції, а також зв’язків налаштування на потрібну операцію КС ярусів конвеєрного операційного пристрою. По конвеєру синхронно з даними під керівництвом тактових імпульсів T по КРК переміщуються і команди. Команда з виходу відповідного регістра налаштовує КС цього яруса на здійснення потрібної операції. В цьому випадку для ярусів конвеєра можна записати: $A_{i+1} = \Phi_i(A_i,K_i)$, де K_i – місткість регістра КРК і-го яруса конвеєра. Перетворення, виконані в такому конвеєрному операційному пристрої над вхідними даними X, можна записати виразом $Y = \Phi_m(K_m,\Phi_m(K_m,\Phi_m(...,(\Phi(K,X))...)))$.

3.5 Таблично-алгоритмічні операційні пристрої

Таблично-алгоритмічні операційні пристрої – це стиснута в таблиці (пам'яті) інформація, яку можна відновити шляхом виконання елементарних та складних операцій.

Арифметичні операції в таблично-алгоритмічних операційних пристроях потрібні для обчислення поправки. Вони базуються на використанні різних методів, зокрема дробово-раціональні наближення, ітераційні процеси, розкладання в ряди, наближення поліномами, ланцюговими дробами тощо. При обчисленні таблично-алгорит-мічним методом аргумент зазвичай розподіляється на 2 частини. Проведення розділення аргументу на більшу кількість частин при неконвеєрній реалізації нераціонально через велику кількість додаткових операцій. При конвеєрній реалізації іноді таке розбиття може виявитися вигідним.

Використання конвеєрного принципу обробки для здійснення арифметичних операцій у таблично-алгоритмічних операційних пристроях дає змогу значно скоротити ємність пам'яті на відміну від прямих табличних методів. Це дозволяє не зменшувати, а й навіть підвищити їх продуктивність. Продуктивність операційного пристрою при використанні таблично-алгоритмічного методу може бути вищою, ніж при застосуванні табличного методу. Це пов'язано з тим, що швидкодія ПЗП залежить від розрядності операндів n, і, у випадку високих вимог до точності, може перевищувати затримку в одному ярусі операційного пристрою, яку можна довести до затримки в ПЗП значно меншого обсягу.

Основна проблема при синтезі таблично-алгоритмічних операційних пристроїв полягає у виборі з усіх відомих методів обчислення даної функції чи їх набору найбільш ефективного та визначення оптимальних співвідношень між обсягом ПЗП і витратами ресурсів обладнання на арифметичну частину при забезпеченні належної продуктивності.

На сьогодні є велика кількість таблично-алгоритмічних методів обчислення елементарних функцій. Розглянемо принципи створення таблично-алгоритмічних операційних пристроїв із застосуванням часткового підходу при обчисленні поправки на прикладі реалізації тригонометричних функцій.

Розбиваючи аргумент на 2 частини X = X1 + X2, утворені його старшими та молодшими розрядами, і користуючись відомими співвідношеннями

$$Sin(X1 + X2) = SinX1CosX2 + CosX1SinX2,$$

$$Cos(X1 + X2) = CosX1CosX2 + SinX1SinX2,$$

можна значно зменшити обсяг ПЗП порівняно з табличним методом. Структуру однотактового операційного пристрою показано на рис. 3.5.

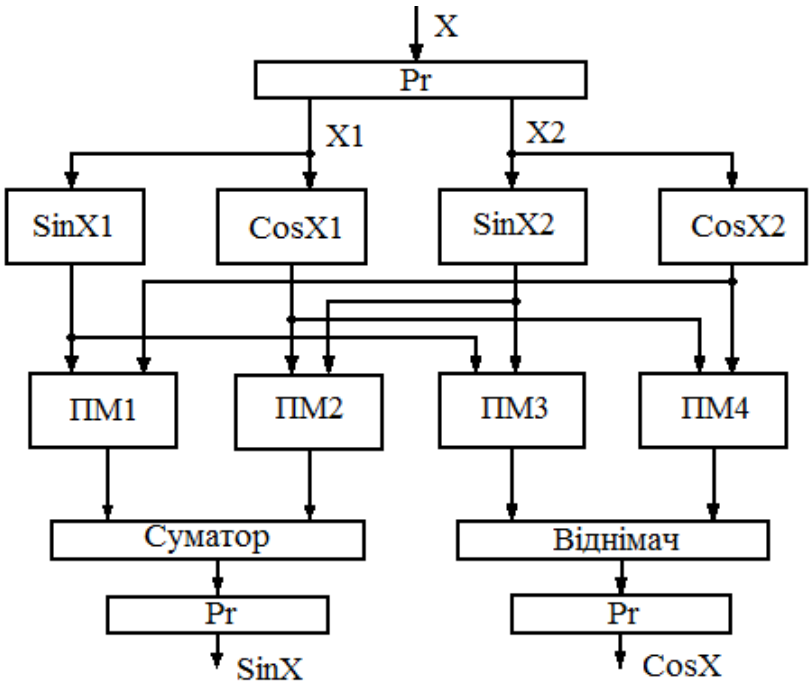


Рисунок 3.5 – Структура однотактового операційного пристрою для обчислення поправки на основі реалізації тригонометричних функцій

Якщо припустити, що X1 – ціле число градусів з дискретністю 1, а X2 – дробове число градусів з дискретністю 0,01, то можна обчислити, що в таблиці, в якій зберігаються SinXl та CosXl, будуть розміщені по 90 15-розрядних слів, а таблиці, що зберігають SinX2 та CosX2 – по 1000 15-розрядних слів. Таким чином, загальний обсяг ПЗП буде складати 5700 бітів, що в 24 рази менше, ніж при обчисленні табличним методом.

Введення в конструкцію конвеєрних регістрів, які можна побудувати за конвеєрним принципом, дозволить підвищити продуктивність описаного пристрою.

Принципи створення таблично-алгоритмічних операційних пристроїв із застосуванням загального підходу для обчислення поправки розглянемо на прикладі обчислення елементарних функцій з використанням раціонального наближення.

Нехай обробці підлягають нормалізовані числа у форматі з ФК. Обрахунок елементарних функцій буде відбуватися на основі методу сегментної апроксимації. Відповідно до цього методу діапазон зміни аргументу [0,5; 1] розподіляється на інтервали з наступним наближенням функції на кожному інтервалі за допомогою виразу $Y = F(X) = A + W(X + B)^2$. A та B, які є константами, обираються за умови мінімізації абсолютної похибки, а константа W вибирається такою, що дорівнює ступеню числа 2 (основа системи числення) та дає змогу замінити операцію множення зсувом. При різних інтервалах константи мають різні значення. Кількість інтервалів та-

кож формується з умови мінімізації абсолютної похибки, причому межі інтервалів формуються k старшими двійковими розрядами аргументу. Це полегшує адресацію пам’яті, в якій записано константи. Структура такого однотактового операційного пристрою наведена на рис. 3.6.

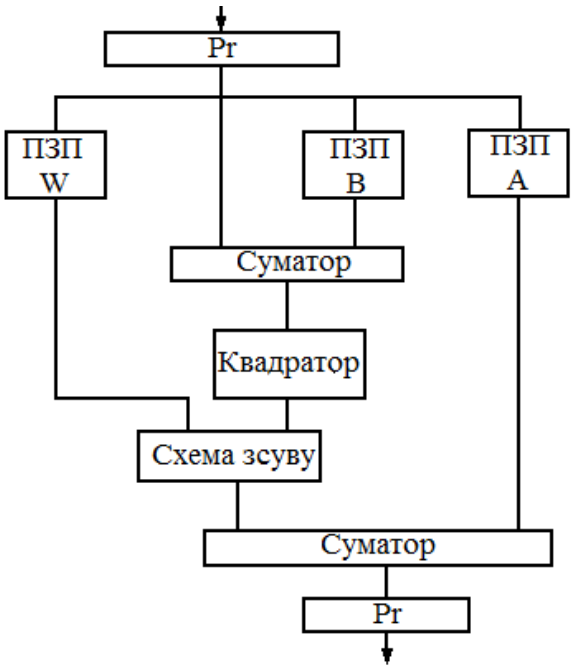


Рисунок 3.6 – Структура однотактового операційного пристрою для обчислення елементарних функцій з використанням раціонального наближення

Реалізація такого пристрою потребує 3 блоки постійної пам’яті для зберігання коефіцієнтів, схему зсуву та 2 суматори. Для 12-розрядних операндів загальний обсяг ПЗП буде 8960 бітів. Константи A, B, W обраховуються для забезпечення заданої точності обчислення. Вихідними даними реалізації алгоритму обрахунку коефіцієнтів є:

- реалізована функція;
- величина інтервалу зміни аргументу;
- розрядність функції та аргументу.

Після обчислення можна отримати такі величини:

- сумарний обсяг ПЗП;
- число підінтервалів зміни аргументу;
- число розрядів, потрібних для кодування номера підінтервалу;
- значення констант A, B, W.

Пристрій, що реалізовує формулу $Y = A + W(X + B)^2$, є багатофункціональним. Константи мають різні значення для різних функцій. Задіяння конвеєра команд дозволяє обраховувати на такому пристрої k функцій від k даних одночасно.

Описані підходи можна застосувати до інших таблично-алгоритмічних методів обчислення елементарних функцій, наприклад, при використанні ланцюгових дробів, раціональних наближень тощо.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Наведіть класифікацію операційних пристроїв.
2. Як працюють табличні операційні пристрої?
3. Поясніть принцип роботи багатотактових операційних пристроїв.
4. Поясніть принцип роботи однотактових операційних пристроїв.
5. Поясніть принцип роботи конвеєрних операційних пристроїв.
6. Поясніть роботу операційного пристрою обчислення елементарних функцій таблично-алгоритмічним методом.

4 ПРИСТРОЇ КЕРУВАННЯ

4.1 Центральний пристрій керування

Пристрій, що здійснює основні функції керування комп'ютером, називається центральним пристроєм керування (ЦПК). Тобто, це сукупність вузлів та блоків процесора, що дають можливість координувати функціонування всіх пристроїв та керувати ними при всіх режимах роботи.

ЦПК реалізує системні та робочі програми, організовує потрібні дії з оцінювання та перетворення вхідної інформації для отримання результату обрахунку.

Вирішення будь-якої задачі зводиться до послідовності вибірки та здійснення команд програмою під керівництвом ЦПК. Таким чином, можна стверджувати, що ЦПК є перетворювачем вхідної командної інформації, що подана командами програми, у вторинну інформацію, а саме: виконавчі адреси та сигнали керування.

До вхідної командної інформації потрібно віднести також коди та сигнали, що описують стан процесора та деяких окремих блоків. Дуже часто в міні- та мікрокомп'ютерах, основною вимогою яких є мінімальна вартість, обладнання комбінують за функціональним призначенням. Наприклад, комбінують пристрої ЦПК і мікропрограмний апарат (МПА) процесора.

Правильне виконання своїх функцій зумовлює такий склад ЦПК:

- регістр команд RGK з полем коду операції (КОП) та полем адреси АДР;
- лічильник адреси команд СТАК;
- автомат керування МПА;
- дешифратор коду операцій DCКОП;
- операційний блок (ОБ), який містить суматор адреси та схеми аналізу режимів роботи, а саме: готовності пам'яті та периферії до обміну інформацією, запитів переривань і прямого доступу до пам'яті;
- інтерфейсні схеми;
- пульт керування «Пульт».

Регістр команд RGK використовується для прийому команди з ОП та зберігання її протягом всього робочого циклу. Залежно від типу машини та складності операції команда має довжину від 1 до 10 і більше байтів.

Залежно від довжини шини вибірки даних за 1 звернення до ОП може прочитуватися вся команда, її частина чи декілька команд.

Лічильник адреси команд СТАК використовується для визначення адреси команди. Після аналізу поточної команди зміст СТАК автоматично збільшується на константу, що дорівнює довжині команди в байтах.

Мікропрограмний автомат МПА1 розшифровує команди та забезпечує виконання сигналами керування програмної частини, а МПА2 – виконання в АЛП мікропрограм операцій. Будова кожного автомата формується на основі схемної чи програмованої логіки. У випадку централізованого управління обидва автомати об’єднуються в один МПА.

Суматор адреси використовується для формування виконавчих адрес операндів та результату операції за інформацією, що знаходиться в коді команди. В загальному випадку виконавчі адреси можна отримати шляхом додавання 3-х компонентів: базової адреси й індексу, що розташовані в блоці РЗП, і коду зміщення в команді.

Пульт керування «Пульт» використовується для керування роботою комп’ютера користувачем. До його складу входить клавіатура, кнопки перемикання та засоби індикації для візуального контролю стану пристроїв і проведення профілактичних дій.

Структурна схема вищезгаданих пристроїв показана на рис. 4.1.

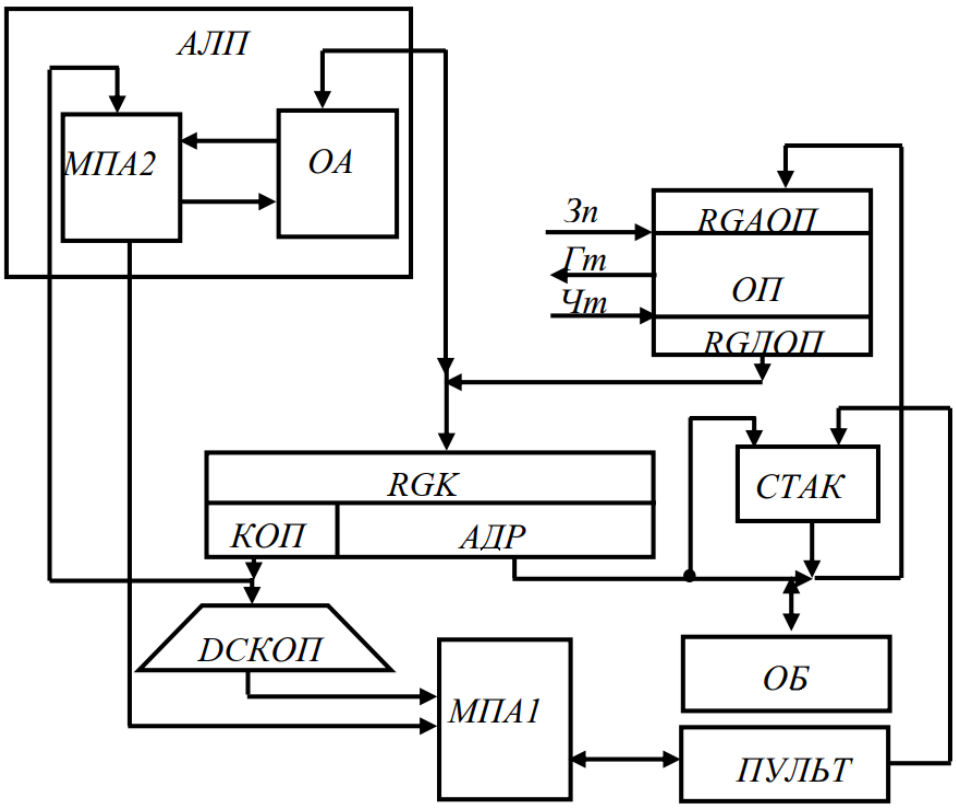


Рисунок 4.1 – Структура взаємодії ЦПК, АЛП і пам’яті

Функціонування комп’ютера відбувається протягом робочих циклів, кожний з яких відповідає виконанню 1 команди програми.

При виконанні кожного робочого циклу в загальному випадку відбувається такі типові дії:

- вибірка з комірки ОП команди, що має виконуватися, та формування адреси наступної команди. При цьому зміст лічильника адреси СТАК надсилається в регістр адреси пам’яті RGAOП. Отриманий за даною адресою код надходить до регістра даних пам’яті, а звідти надсилається до регістра команд RGK, після чого зміст лічильника адреси СТАК збільшується на константу, яка є довжиною команди в байтах;
- формування адрес виконання та зчитування за ними операндів з ОП. Зміст адресної частини команди надходить до ОБ, де генеруються виконавчі адреси. Зчитані операнди приходять до ОА, де зберігаються в блоці РЗП;
- розшифрування коду операції в МПА2 здійснюється послідовністю мікрооперацій, які визначаються мікропрограмою даної операції та записують результати операції в пам’ять або в РЗП;
- формування в МПА2 сигналу АЛП про кінець операції. Перехід до п. 1.

4.2 Пристрій керування з жорсткою логікою

Стандартну структурну схему пристрою керування з жорсткою логікою наведено на рис. 4.2.

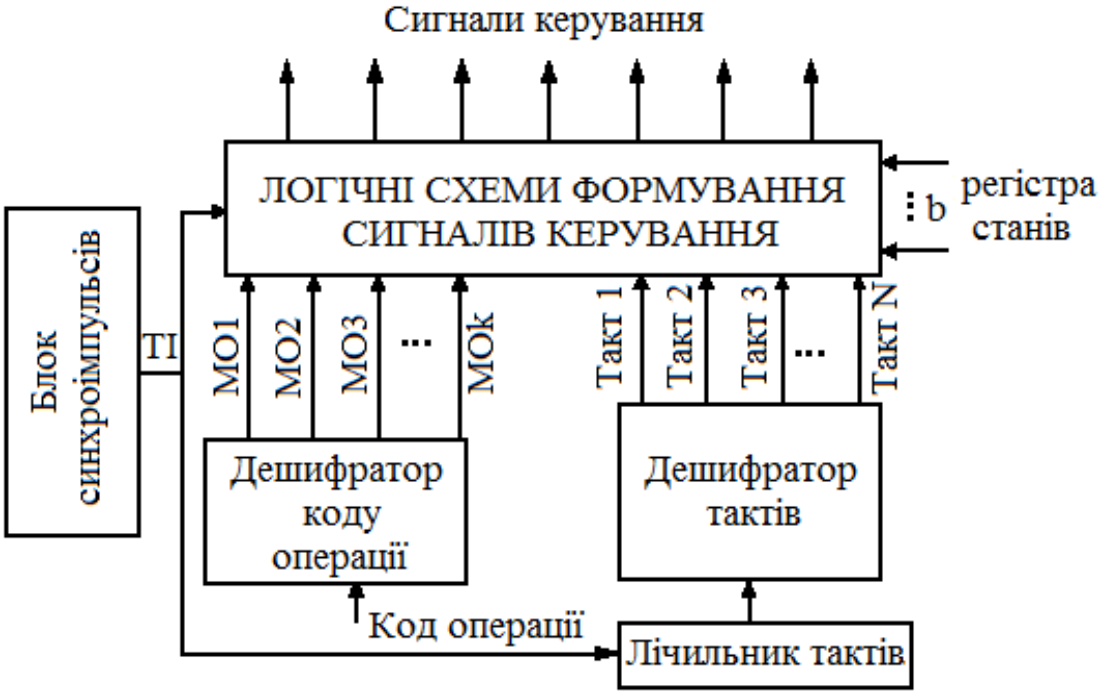


Рисунок 4.2 – Стандартна структурна схема пристрою керування з жорсткою логікою

Пристрій керування складається з:

- блоку синхроімпульсів, що генерує тактові імпульси ТІ, які потрібні для синхронізації роботи пристрою керування;
- лічильника тактів, який містить номер такту, що виконується в даний момент;
- дешифратора коду операції та дешифратора тактів, що трансформують двійковий код в однорядний;
- логічних схем формування сигналів керування.

Дешифратор коду операції з регістра команд генерує сигнал активізації мікрооперації МО на відповідній шині. З кожним тактом до лічильника тактів додається +1 сигнал із блоку синхроімпульсів. Дешифратор тактів формує сигнали, які відповідні поточному такту.

Логічні схеми формування сигналів керування формують сигнали керування для виконання потрібних в даному такті мікрооперацій.

Крім вищенаведених компонентів пристрою керування, також до його складу входить контролер послідовності сигналів керування, до якого надходять тактові імпульси з блоку синхроімпульсів і також код робочого режиму комп'ютера. Він має 2 окремих режими роботи: звичайний режим і режим запуску комп'ютера. Контролер послідовності сигналів керування є ядром пристрою керування.

4.2.1 Пристрій керування на основі синхронних елементів часової затримки

Пристрій керування на основі синхронних елементів часової затримки формується з блок-схеми, яка дозволяє вказати порядок формування послідовності сигналів керування. Таким чином, структура побудованого пристрою керування фактично повторює структуру блок-схеми, тобто схема пристрою керування ілюструє послідовність формування сигналів керування. В основі методу побудови пристрою керування на основі синхронних елементів часової затримки лежить те, що створення набору сигналів керування в послідовні моменти часу можна здійснювати за допомогою їх часової затримки. Тобто, маючи сигнал Сі в момент часу tl, за допомогою його затримки на 1 такт, можна сформувати сигнал С2 в момент часу t2 і т. д.

Незважаючи на простоту проектування пристрою керування на основі синхронних елементів затримки, існує недолік, який полягає в тому, що число потрібних схем затримки дорівнює числу станів n. Також існує проблема синхронізації багатьох розподілених елементів затримки.

4.2.2 Пристрій керування на основі лічильників

В основі побудови пристрою керування на основі лічильників знаходиться часова діаграма роботи комп'ютера, що показує зміну в часі кожного сигналу керування. Наприклад, на рис. 4.3 наведено частину часової діаграми роботи комп'ютера, де ТІ – тактові імпульси, які надходять з блоку синхроімпульсів, С1–С5 – частина сигналів керування, що генеруються пристроєм керування.

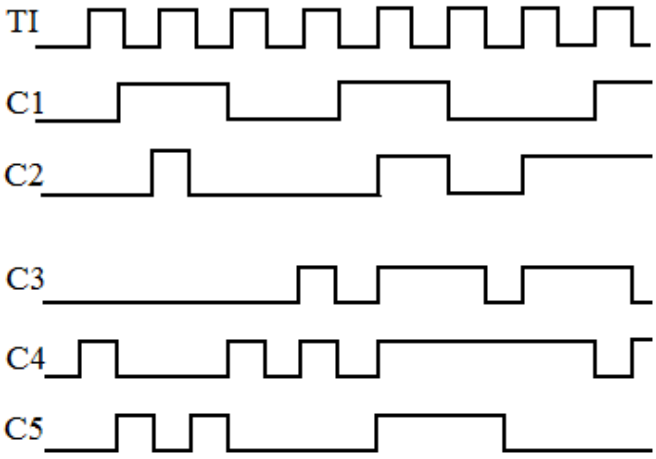
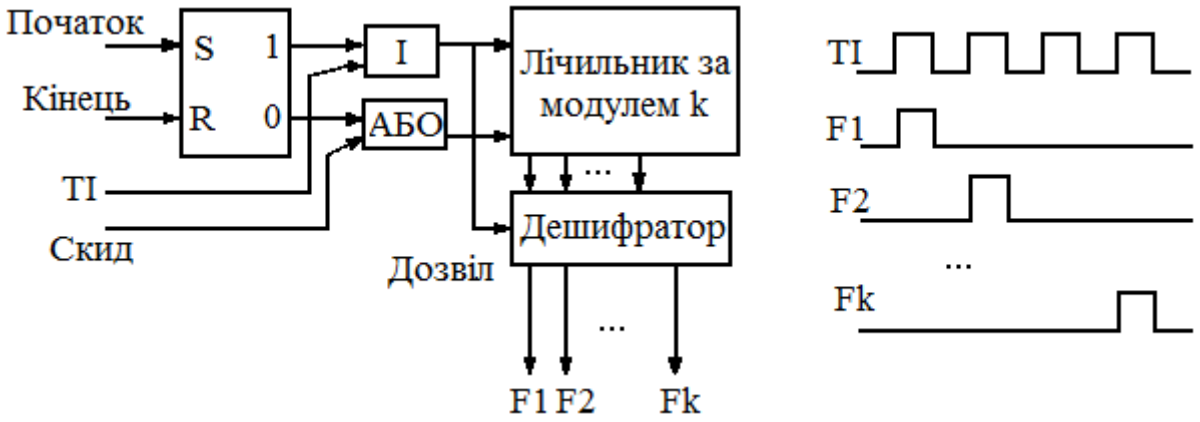


Рисунок 4.3 – Частина часової діаграми роботи комп’ютера

Основним елементом пристрою керування на основі лічильників є лічильник за модулем k, у якого виходи з’єднані з дешифратором (рис. 4.4, а).



а б

Рисунок 4.4 – Генератор послідовності одноімпульсних сигналів:
а – структурна схема;
б – часова діаграма сигналів на виході

Коли на вхід лічильника за модулем k надходять тактові імпульси, він рахує їх від 0-го до k-го, після чого цикл починається знову. Таким чином, в результаті на виході дешифратора буде генеруватися послідовність одноімпульсних сигналів Fl, F2, ..., Fk. Їх часова діаграма наведена на рис. 4.4, б. Кожен сигнал має одиничне значення лише протягом 1 тактового періоду. Таким чином, час 1 циклу роботи лічильника поділено на k рівних частин. 2 додаткових вхідних сигнали початку та кінця роботи та тригер RS типу забезпечують створення сигналів дозволу роботи лічильника та його скиду. Базова частина схеми пристрою керування на основі лічильника має вигляд, наведений на рис. 4.5.

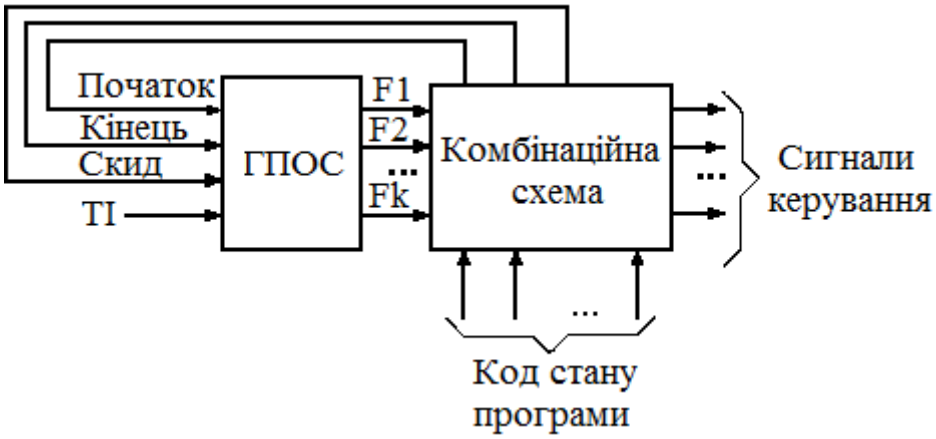


Рисунок 4.5 – Структура пристрою керування
на основі лічильника

Кожен сигнал Fi (i = 1, 2, …, k) на виході генератора послідовності одноімпульсних сигналів ГПОС активізує певний набір сигналів керування на виході комбінаційної схеми в кожному такті виконання команди комп'ютером, з урахуванням коду стану програми. Доцільність застосування лічильника за модулем можна пояснити циклічністю сигналів керування вузлами комп'ютера.

Потрібно зазначити, що лічильник за модулем k можна використати й в схемі пристрою керування на основі синхронних елементів часової затримки замість k послідовно з'єднаних тригерів.

4.3 Пристрій мікропрограмного керування

Пристрій мікропрограмного керування генерує послідовність сигналів, які потрібні для здійснення програми на комп'ютері. Програма формується з певної послідовності команд. Команда в комп'ютері виконується за 1 чи за кілька тактів, в кожному з яких виконується 1 чи декілька мікрооперацій. Кожна мікрооперація є деякою елементарною дією передавання чи перетворення інформації, що ініціюється надходженням сигналу керування (мікронаказу) на вхід керування відповідного пристрою. Послідовність елементарних мікронаказів, що формуються в 1 такті пристроєм керування, називається мікрокомандою. Послідовність мікрокоманд, яку потрібно здійснити задля здійснення 1 команди, називається мікропрограмою. Звичайно, мікропрограма може складатися й лише з 1 мікрокоманди.

Основними принципами побудови пристрою мікропрограмного керування є:

1. Всі мікронакази, що мають бути виконані в 1 такті роботи комп'ютера, збираються в 1 слово керування, що називається мікрокомандою;
2. Кожній команді з системи команд комп'ютера ставиться у відповідність послідовність мікрокоманд, які потрібні для її виконання, тобто мікропрограма здійснення команди в комп'ютері;

3. Всі мікрокоманди зберігаються в пам’яті. Це може бути основна пам’ять комп’ютера, але для зберігання мікрокоманд в більшості комп’ютерів задіяна окрема пам’ять, яка називається пам’яттю мікрокоманд;

4. Реалізація певної команди вимагає зчитування з пам’яті мікрокоманд відповідну послідовність мікрокоманд (мікропрограму) і надсилати розподілену в часі послідовність сигналів керування на відповідні входи керування вузлів комп’ютера.

На рис. 4.6 наведено основний елемент пристрою мікропрограмного керування – пам’ять мікрокоманд – та вузли на її входах і виходах, а саме: мікропрограмний лічильник (МКПЛ), який використовується для зберігання адреси мікрокоманди, та регістр мікрокоманди (РгМК).

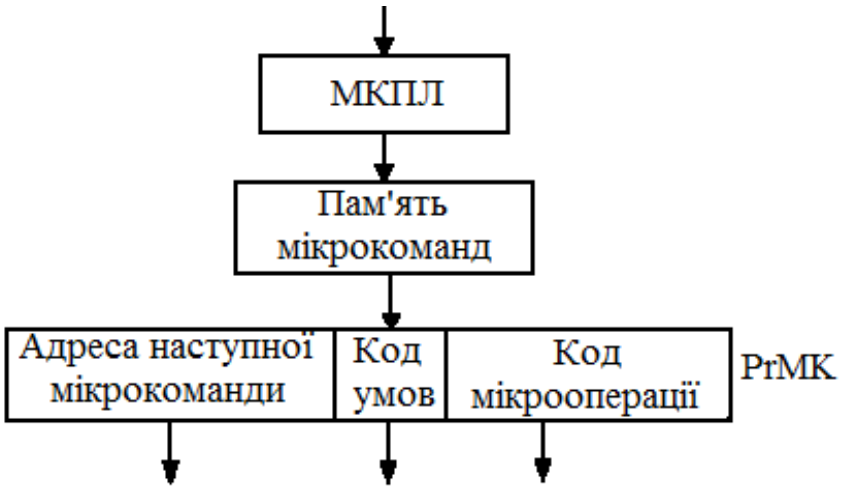


Рисунок 4.6 – Пам’ять мікрокоманд з регістром адреси мікрокоманди на вході та регістром мікрокоманди на вході

На рис. 4.6 наведено й формат самої мікрокоманди, який містить такі поля: код мікрооперації, на основі якого генеруються мікронакази, що виконуються за 1 такт роботи комп’ютера; код умов, що вказує, за яких умов буде змінено послідовність зчитування мікрокоманд з пам’яті, а також адреса наступної мікрокоманди.

Структуру пристрою мікропрограмного керування наведено на рис. 4.7.

В регістрі команди (РгК) зберігається команда, що виконується комп’ютером. За її КОП з пам’яті адрес перших мікрокоманд мікрокодів зчитується адреса пам’яті мікрокоманд, в якій міститься перша мікрокоманда з сукупності мікрокоманд (мікрокоду) її виконання. Зазвичай пам’ять мікрокоманд організується на основі ПЗП, хоча її також можна реалізувати й на основі ОЗП, особливо на етапах налаштування комп’ютера. В цій пам’яті зберігаються мікрокоманди для всіх команд комп’ютера, а також для початку роботи комп’ютера та для обробки переривань. В схемі фор-

мування адреси (СФА), до складу якої входять пам’ять адрес перших мікрокоманд мікрокодів, суматор-мультиплексор (СМ-МП), МКПЛ і КС формування переходу, знаходиться адреса комірки пам’яті мікрокоманд, в якій міститься наступна мікрокоманда. Ця адреса створюється з урахуванням полів адреси наступної мікрокоманди (АНМК) та коду умови з регістра мікрокоманди, а також сигналів стану, які надходять з регістра слова стану програми. Мікрокоманда зчитується з пам’яті в РгМК, де вона зберігається протягом 1 такту роботи комп’ютера. Базуючись на основі коду мікрооперації (КМКО) з регістра мікрооперації на виході дешифратора мікрооперацій (ДШМКО) генеруються сигнали керування.

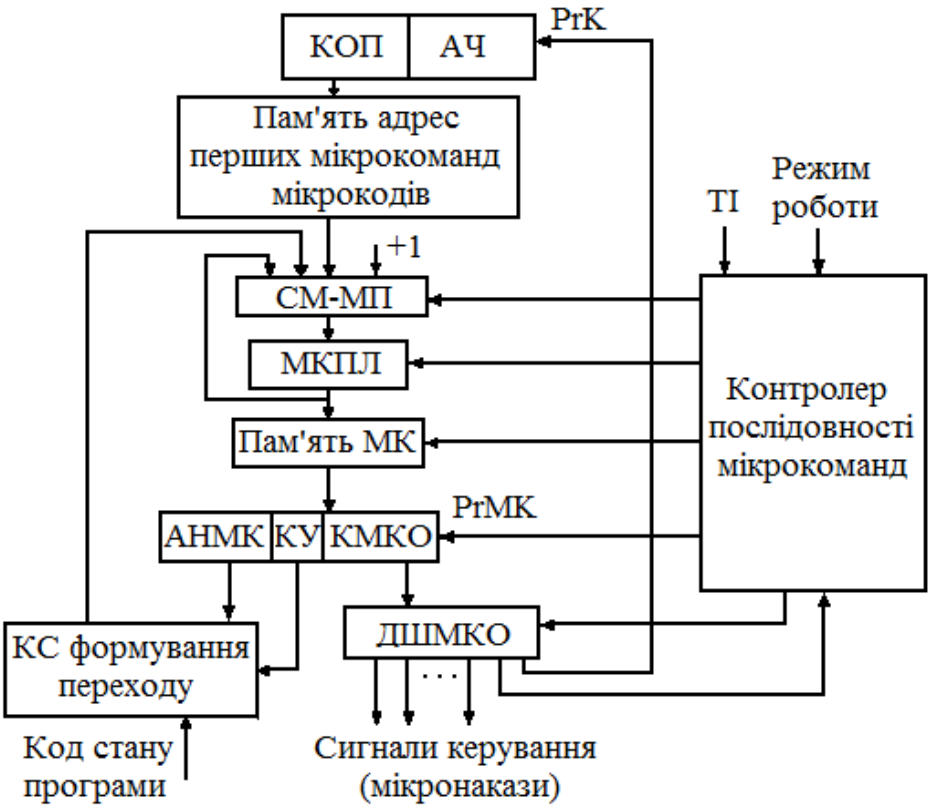


Рисунок 4.7 – Структура пристрою мікропрограмного керування

Компонент пристрою керування синхронізується з контролером послідовності мікрокоманд, до якого ззовні надходять тактові імпульси з генератора тактових імпульсів і код режиму роботи комп’ютера. Контролер послідовності мікрокоманд являє собою ядро пристрою керування. Він містить 2 окремих режими роботи: звичайний режим і режим запуску комп’ютера.

В звичайному режимі контролер послідовності мікрокоманд створює сигнали керування завдяки роботі пристрою керування. Вхідні тактові імпульси забезпечують його часову синхронізацію, що дає змогу створювати такі послідовності сигналів:

1. Керівництво процесом формування адреси мікрокоманди. Ця адреса отримується шляхом:

■ запису в МКПЛ адреси першої мікрокоманди мікропрограми виконання відповідної команди з пам’яті адрес перших мікрокоманд мікрокодів;

■ запису в МКПЛ адреси наступної мікрокоманди мікропрограми виконання відповідної команди після виконання в СМ-МП операції приросту вмісту МКПЛ на 1;

■ запису в МКПЛ АНМК з адресного поля РгМК, з урахуванням коду умови (КУ) з поля умовного переходу РгМК, та коду стану програми з регістра слова стану програми, що відбувається в КС формування переходу.

2. Керівництво зчитуванням мікрокоманд з пам’яті мікрокоманд за адресами з МКПЛ та їх записом до РгК.

3. Стимулювання ДШМКО, що здійснює дешифрування КМКО з відповідного поля РгМК і створює сигнали керування (мікронакази).

Кожна команда процесора здійснює виконання відповідної мікропрограми. У випадку. коли пристрій керування завершує виконання мікропрограми однієї команди комп’ютера, про що його інформує контролер послідовності мікрокоманд сигналом з виходу ДШМКО, він обирає з основної пам’яті наступну команду та здійснює запис її в РгК сигналом із виходу ДШМКО, після чого починає виконувати її шляхом зчитування з пам’яті мікрокоманд наступної мікропрограми.

В процесі виконання різних команд використовуються загальні ділянки мікропрограм, що називаються мікропідпрограмами.

В режимі запуску комп’ютера пристрій керування встановлює вміст різних регістрів комп’ютера в початковий стан, шляхом скидання чи запису до них певних конкретних значень. Після цього він записує до програмного лічильника ПЛ апаратно генеровану адресу та починає виконувати програму. Для окремих комп’ютерів апаратно згенерована адреса – це вектор скидання, який є адресою першої команди комп’ютера, що виконується після старту. Для інших комп’ютерів апаратно генерована адреса – це адреса вектора скидання, який є адресою першої команди комп’ютера, що виконується після старту. В даному випадку пристрій керування спочатку обирає з основної пам’яті вектор скидання та записує його до програмного лічильника ПЛ.

4.3.1 Організація мікропрограм в пам’яті мікрокоманд

Існує багато шляхів можливої організації мікропрограм в пам’яті мікрокоманд. Один з них передбачає послідовне розміщення мікрокоманд в пам’яті. Кожна машинна команда має власну послідовність мікрокоманд в пам’яті мікрокоманд. Додатково пам’ять мікрокоманд має мікрокоманди

для проведення «вибірки» команд з основної пам’яті, запуску переривання та деяких інших дій, спрямованих на керування.

Розглянемо цей тип організації мікропрограм в пам’яті мікрокоманд. Після того, як пристрій керування обирає машинну команду з основної пам’яті та поміщає її в РгК, він генерує адресу точки входу для коду операції команди КОП, що є адресою першої мікрокоманди мікропрограми. Наприклад, у випадку, якщо пристрій керування обирає мікрокоманду для коду операції КОП1, він генерує адресу А1 (рис. 4.8). Формування адреси точки входу здійснює пам’ять адрес перших мікрокоманд мікрокодів блоку обчислення адреси (рис. 4.7).

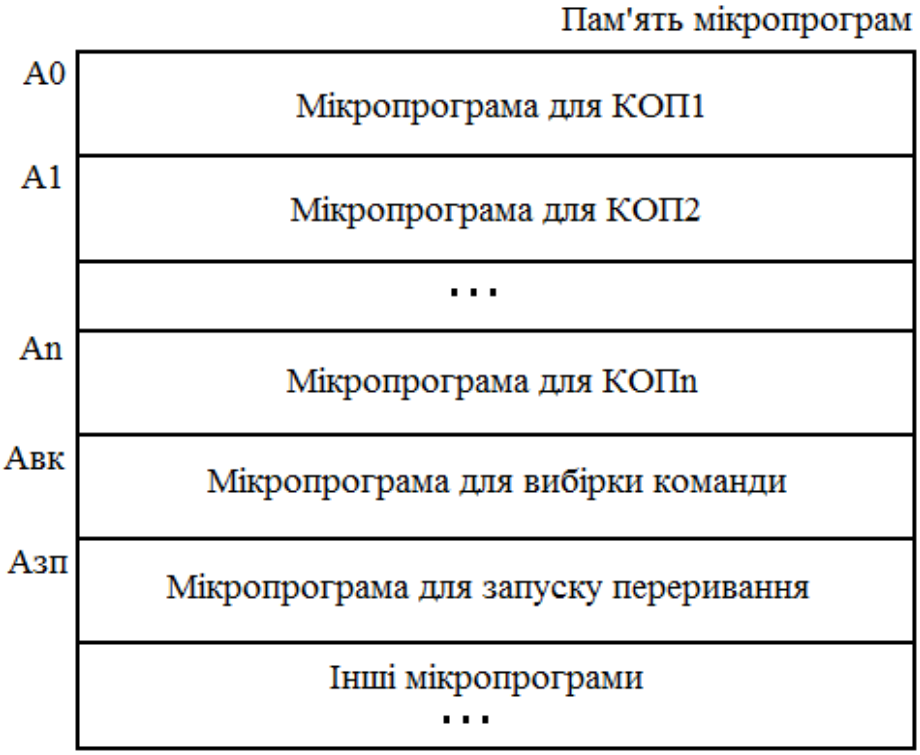


Рисунок 4.8 – Послідовне розміщення команд в пам’яті

Після формування адреси точки входу пристроєм керування контролер послідовності мікрокоманд збільшує вміст МКПЛ задля отримання адреси кожної наступної мікрокоманди.

Після виконання останньої мікрокоманди в мікропрограмі (тобто, після закінчення здійснення однієї машинної команди) контролер послідовності мікрокоманд ще раз виконує мікропрограму «вибірки» наступної команди з основної пам’яті. На рис. 4.8 це перехід до комірки Авк пам’яті мікрокоманд. Існують мікрокоманди переходу, що мають адресу переходу та мікронакази для СФА додатково до мікронаказів для інших вузлів комп’ютера. Контролер послідовності мікрокоманд разом з СФА застосовує АНМК та КУ переходу для знаходження адреси наступної мікрокоманди.

Для керівництва переходом в КМКО кожної мікрокоманди існує інформація про належність цієї мікрокоманди до мікрокоманд переходу. Таким чином, в кожному такті на виході ДШМКО формується мікронаказ, що надає інформацію контролеру послідовності мікрокоманд про належність або неналежність цієї мікрокоманди до мікрокоманд переходу. Якщо ця мікрокоманда є мікрокомандою переходу, то контролер послідовності мікрокоманд інформує схему формування адрес про потребу формування адреси переходу, а не приріст на 1 вмісту МКПЛ. При цьому, якщо це безумовний перехід, то перехід здійснюється за адресою з адресного поля мікрокоманди. Якщо ж це умовний перехід, то адреса наступної мікрокоманди створюється з урахуванням коду умов переходу та коду стану програми, що надходить з регістра стану програми регістрової пам'яті процесора.

4.3.2 Горизонтальне та вертикальне мікропрограмування

Вибір формату мікрокоманди є досить складним. З одного боку, мікрокоманда має містити коди керування вузлами комп'ютера, забезпечувати формування послідовності мікрокоманд в мікропрограму та можливість змінювати порядок мікрокоманд в мікропрограмі. З іншого боку, ці функції мають здійснюватись з мінімальною кількістю бітів в мікрокоманді, з мінімальною кількістю слів з пам'яті мікрокоманд та з мінімальним часом на виконання мікропрограми.

Мікропрограмування за способом формування сигналів керування поділяється на горизонтальне та вертикальне.

При горизонтальному мікропрограмуванні кожний розряд поля коду мікрооперації мікрокоманди формує 1 сигнал керування (мікронаказ) для відповідного входу керування функціонального вузла комп'ютера (рис. 4.9).

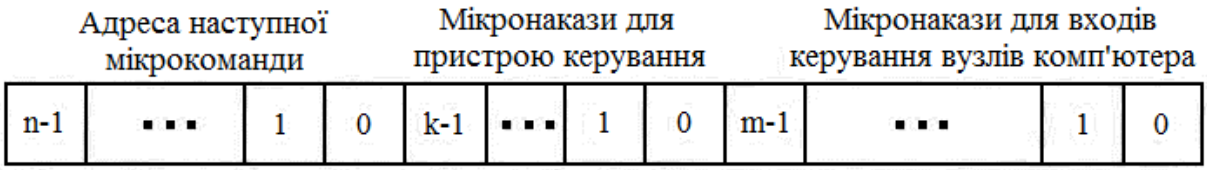


Рисунок 4.9 – Формат мікрокоманди
при горизонтальному мікропрограмуванні

Розряди мікрокоманди визначаються з виразу $M = n + k + m$, де n – розрядність адреси мікрокоманди, причому $n = \log_2 N$, де N – кількість мікрокоманд в пам'яті, k – кількість мікронаказів, потрібних для керування вузлами пристрою керування, m – кількість мікронаказів, потрібних для керування вузлами комп'ютера.

Оскільки в комп’ютерах кількість мікронаказів може досягти декількох сотень, то мікрокоманда в цьому випадку стає дуже широкою.

Кількість бітів керування мікрокоманди можна зменшити, використовуючи нижчевказані способи.

■ *Групування бітів*. Формуються групи мікронаказів, що завжди виконуються одночасно. При цьому для групи резервується лише 1 біт.

■ *Групування форматів*. Формуються групи мікронаказів, з яких в цьому такті виконується лише 1. Ці мікронакази кодуються в полі, де кількість бітів $k = \log_2 L$, де L – кількість мікронаказів.

■ *Групування мікронаказів*. Для групи мікронаказів резервується 1 біт мікрокоманди та використовується багататактова синхронізація.

При вертикальному мікропрограмуванні мікрокоманда складається з полів коду мікрооперації, коду умов переходу та адреси наступної мікрокоманди, про що було вказано вище (див. рис. 4.6). Таким чином, формат мікрокоманди подібний формату команди комп’ютера. Вертикальне мікропрограмування дає змогу більш ефективно використовувати поля мікрокоманди, що дозволяє робити команди коротшими, а ємність пам’яті меншим порівняно з горизонтальним мікропрограмуванням. При цьому горизонтальне мікропрограмування є швидшим процесом, оскільки не потребує використання дешифраторів.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Вкажіть призначення пристрою керування.
2. Вкажіть призначення блоку синхронізації.
3. Вкажіть призначення дешифратора коду операції.
4. Вкажіть призначення дешифратора тактів.
4. На основі чого здійснюється побудова пристрою керування?
5. Наведіть структуру пристрою мікропрограмного керування та дайте пояснення організації його роботи.
6. Яким чином використовують тактовані елементи часової затримки при створенні пристрою керування?
7. Яким чином використовують лічильники при створенні пристрою керування?
8. Як можна побудувати часову діаграму роботи комп’ютера?
9. Які основні принципи будови пристрою мікропрограмного керування?
10. Наведіть формат мікрокоманди.

5 ПРОГРАМОВАНІ ЛОГІЧНІ МАТРИЦІ ТА ПРОГРАМОВАНІ ЛОГІЧНІ ІНТЕГРАЛЬНІ СХЕМИ

5.1 Програмовані логічні матриці

5.1.1 Пристрої на основі програмованих логічних матриць

Принцип розробки пристроїв на мікросхемах програмувальної логіки вперше був реалізований у вигляді програмованих логічних матриць (ПЛМ), які є першими представниками універсальних програмованих логічних інтегральних схем (ПЛІС).

Конструкція ПЛМ базується на тому, що будь-яку логічну функцію можна подати у вигляді диз'юнктивної нормальної форми (ДНФ). ДНФ є сумою кон'юнктивних членів, які перетворюють логічну функцію в 1, та може бути організована на основі логічних вентилів 3-х типів: «І» (and), «АБО» (or), «НІ» (not). Диз'юнктивна нормальна форма функції Y (таблиця істинності 5.1) має вигляд

$$Y = \overline{X_1 X_2 X_3} + X_1 X_2 \overline{X_3} + X_1 \overline{X_2} X_3 .$$

Таблиця 5.1 – Таблиця істинності

X3	X2	X1	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Як видно, у внутрішній структурі ПЛМ знаходяться лінії з прямими та інвертованими вхідними сигналами, що їх можна надіслати на входи будь-якого вентиля (and), вихід якого можна приєднати до входу вентиля (or).

Параметрами ПЛМ (рис. 5.1) є кількість вхідних сигналів m, число термів l і число виходів n.

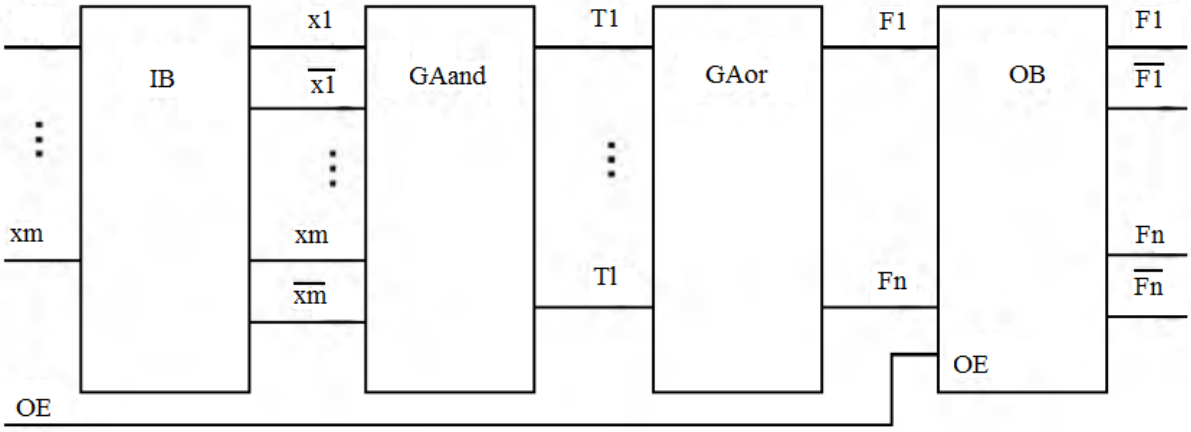


Рисунок 5.1 – Структура ПЛМ

Вхідні буфери IB створюють сигнали потрібної потужності для живлення матриці GAand і перетворюють вхідні сигнали x у парафазні. З виходу IB прямі та інвертовані вхідні сигнали надходять на входи кон'юнктерів матриці GAand, на виході якої створюються l термів T (кон'юнктивних членів). Число l сформованих термів дорівнює числу кон'юнкторів у матриці GAand. Потім терми надходять на входи матриці GAor, тобто на входи диз'юнкторів, що створюють вихідні функції. Число n створених функцій F дорівнює числу диз'юнкторів у матриці GAor.

Сигнали з виходів матриці Geor приходять у блок виведення OB, що створює прямі й інверсні сигнали вихідних функцій F та забезпечує потрібну навантажувальну здатність виходів. Сигнал керування OE дає дозвіл чи забороняє вихід ПЛМ на зовнішні шини. Отже, за допомогою ПЛМ можна створити систему n логічних функцій від m аргументів, що має не більше l термів.

ПЛМ реалізується на біполярній технології та на МДН-транзисторах. У матрицях існують системи горизонтальних і вертикальних зв'язків. Під час програмування у вузлах перетину знищуються чи утворюються елементи зв'язку. В першому випадку, під час виробництва ПЛМ утворюються всі зв'язки в місцях перетину провідників, а під час програмування зайві зв'язки ліквідовуються. В другому випадку, специфікація зв'язків відбувається за допомогою додатково створеної маски – фотошаблона. Наприклад, на рис. 5.2 наведено структурну схему ПЛМ з 3-ма входами та 2-ма виходами до програмування (відсутність перемички в матрицях GAand та GAor).

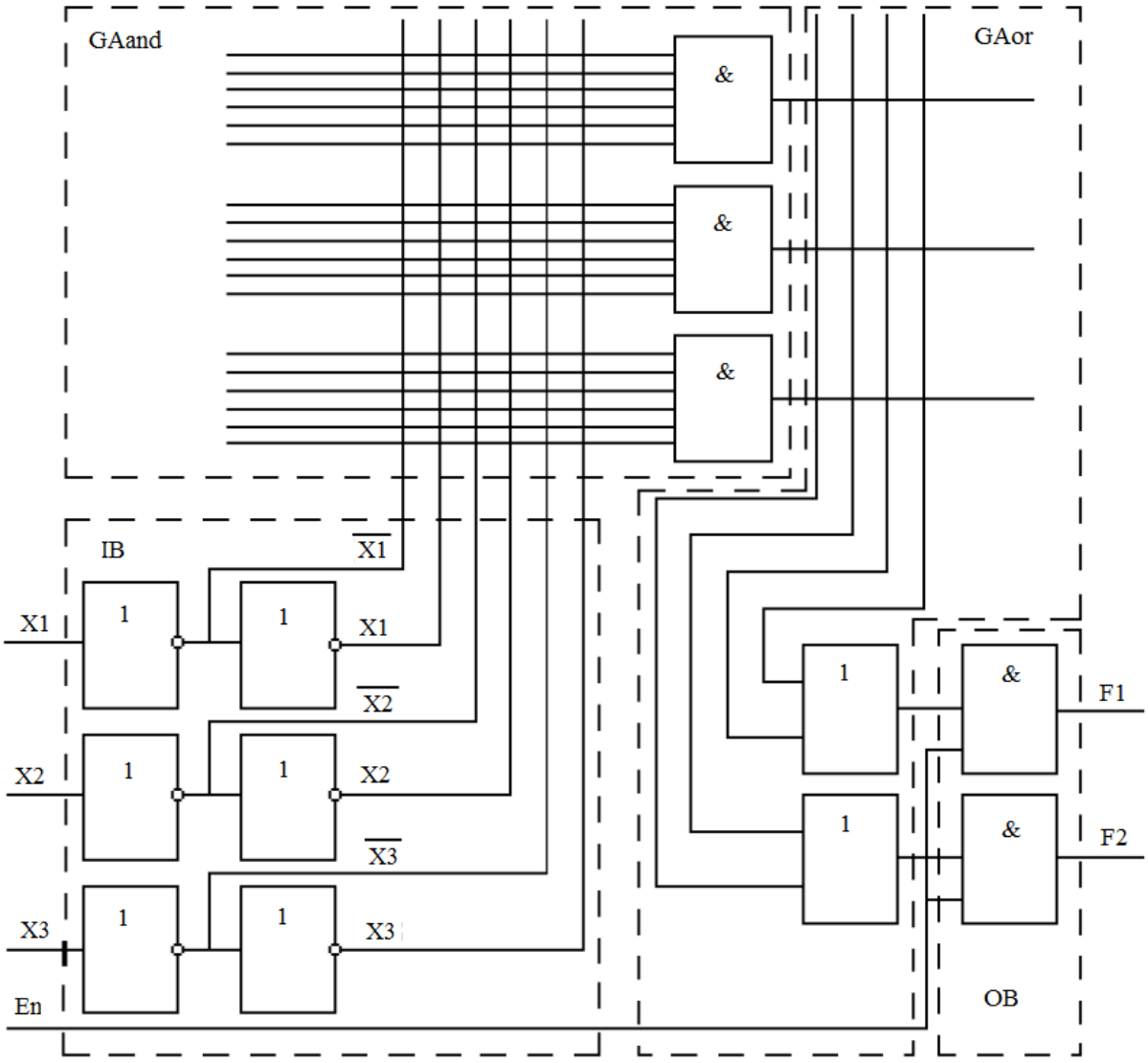


Рисунок 5.2 – Структурна схема ПЛМ
до програмування

Для реалізації функцій $F1 = \overline{X_1 X_2 X_3} + X_1 \overline{X_2 X_3}$, $F2 = X_1 X_2 X_3 + X_1 \overline{X_2 X_3}$ створюються відповідні перемички у матрицях GAand і GAor (рис. 5.3).

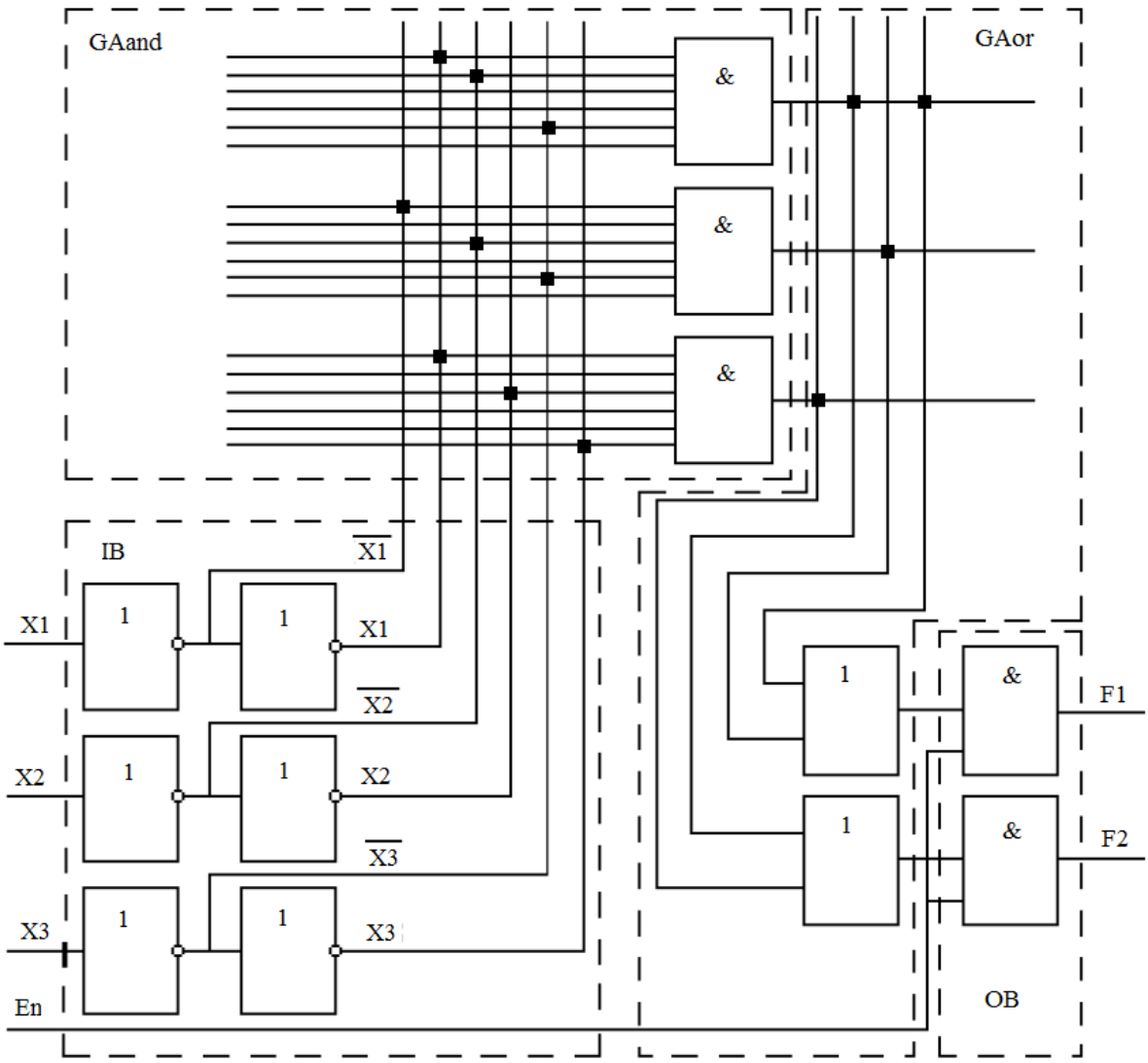


Рисунок 5.3 – Структурна схема ПЛМ
після програмування

Прикладами таких ПЛІС можуть бути вітчизняні ІС К556РТ1, РТ2, РТ21.

5.1.2 Пристрої на основі програмованої матричної логіки

Під час вирішення найбільш типових практичних завдань логічної потужності ПЛМ часто застосовується не в повній мірі (системи перемикальних функцій не містять великих перетинів за однаковими термами). В цих випадках використання виходів будь-яких кон'юнкторів будь-якими диз'юнкторами є зайвим ускладненням схемотехнічної реалізації. Структури, в яких виходи елементів and жорстко зв'язані з елементами or, є програмованою матричною логікою (ПМЛ або PAL). Порівняно з ПЛМ, структури ПМЛ є менш гнучкими функціонально, з огляду на фіксовані матриці GAor, але їх створення та використання спрощується. Приклад структурної схеми ПМЛ, яка містить m входів і n виходів, показано на рис. 5.4.

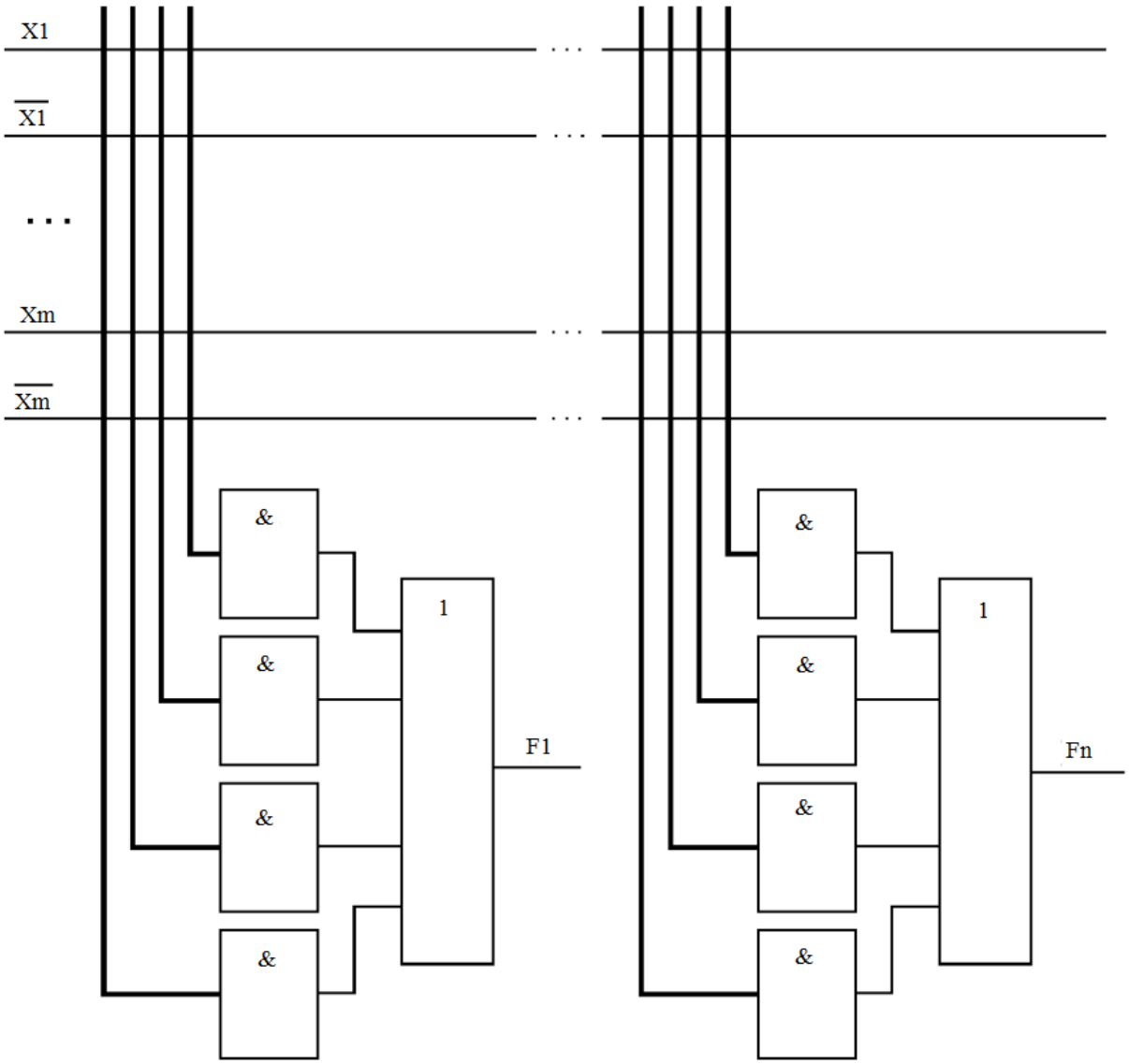


Рисунок 5.4 – Структурна схема ПЛМ

Як видно з цієї схеми, є 4n елементів and, оскільки кожному елементу or надається структура з 4-х кон'юнкторів). Інтегральні схеми середнього ступеня інтеграції з ПМЛ виготовляються низкою таких фірм-виробників, як INTEL, ALTERA, AMD, LATTICE тощо.

5.1.3 Пристрої на основі складних програмованих логічних пристроїв

Наведені архітектури ПЛІС мають мало осередків і використовуються для створення відносно простих пристроїв. Наступним кроком в розвитку архітектури ПМЛ на сьогодні є складні програмовані логічні пристрої (СПЛП). Вони містять декілька матричних логічних блоків (МЛБ або MLB), об'єднаних комутаційною матрицею (КМ або PIA). Кожен МЛБ є структурою типу ПМЛ-програмованою матрицею GAand та фіксованою матрицею GAor. Структурну схему СПЛП наведено на рис. 5.5.

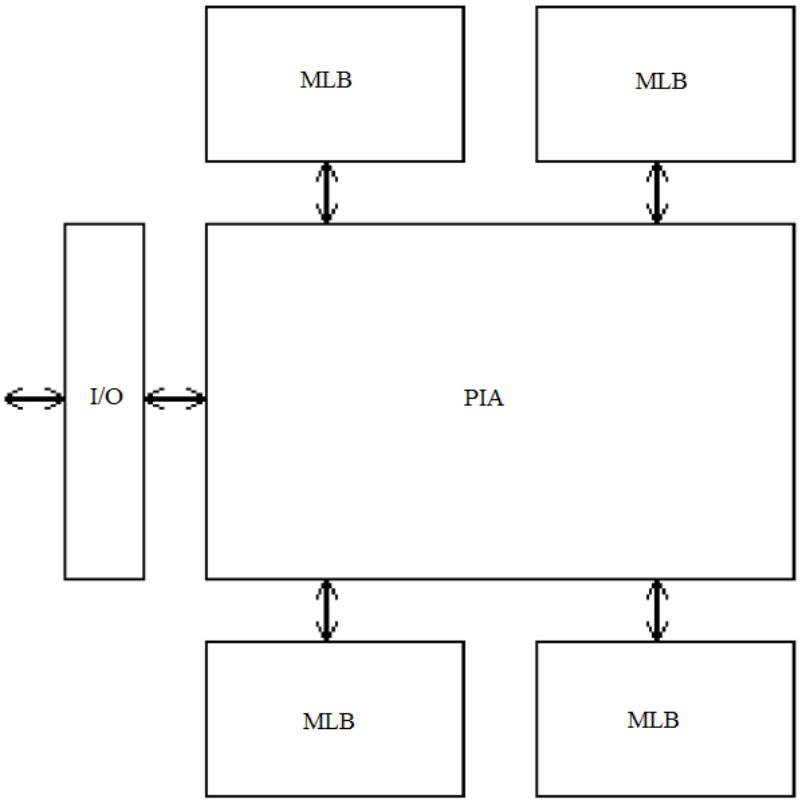


Рисунок 5.5 – Структурна схема СПЛП

Як перемички в матриці міжз'єднань застосовуються МДН-транзистори з плаваючим затвором. Ці ПЛІС виготовляються за технологією Flash-пам'яті.

У СПЛП (CPLD) застосовується безперервна система ідентичних зв'язків, що дозволяє передбачувати затримку сигналів у провідниках. Це дозволяє істотно полегшити проектування та виготовлення працездатних схем високої швидкодії. ПЛІС типу CPLD мають високий ступінь інтеграції (до 12000 вентилів, до 560 макроосередків). Відомими пристроями з архітектурою СПЛП є мікросхеми MAX 5000 та MAX 7000 фірми ALTERA. Вони використовуються для створення цифрових автоматів, інтерфейсних схем і схем керування тощо.

5.1.4 Пристрої на базових матричних кристалах

Альтернативою ПЛІС на ПЛМ є архітектура пристроїв на базових матричних кристалах (БМК або GA). Їх основою БМК є сукупність на кристалі базових осередків (БО) – наборів елементів схем, які постійно повторюються на площі кристала та займають центральну ділянку. На периферії розміщені БО введення/виведення, які орієнтовані на створення зовнішніх зв'язків БМК. Отже, БМК є кристалом-заготовкою, що його можна перетворити в потрібну схему створення відповідних з'єднань. Споживач має змогу реалізовувати на основі БМК велику кількість пристроїв певного класу, задаючи для кристала той чи інший варіант рисунку міжз'єднань компонентів.

В процесі проектування БМК збалансовують оптимальним чином число БО, трасувальні ресурси кристала та число контактних майданчиків для підключення зовнішніх висновків. Трасувальні можливості БМК визначаються площею, яка відводиться для міжз'єднувальних зв'язків в ортогональних напрямках, і кількістю шарів міжз'єднань. Недостатня трасувальна здатність призводить до зменшення кількості задіяних у ході створення схеми БО. Надлишкова трасувальна здатність приводить до нераціонального використання площі кристала, що знижує рівень інтеграції БМК та підвищує його вартість.

Потрібно уточнити термінологію, яка використовується під час проектування БМК.

БО внутрішньої ділянки БМК є матричними базовими осередками (МБО).

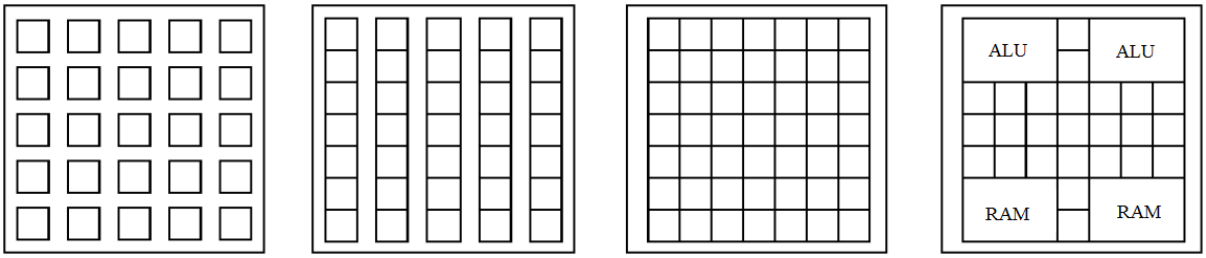
БО периферійної ділянки кристала називаються периферійними базовими осередками (ПБО).

Використовуючи елементи МБО можна сформувати 1 логічний елемент, а реалізація складних функцій вимагає використання декількох осередків. При цьому з елементів МБО можна сформувати будь-який функціональний вузол, а склад елементів осередка характеризується схемою найскладнішого вузла.

Функціональний осередок (ФО) – функціонально закінчена схема, яка реалізується шляхом з'єднання елементів у межах однієї чи декількох БО.

Канали трасування – ділянки на БМК для можливого розміщення міжз'єднань.

За принципом розміщення БО БМК можна поділити на: канальні, безканальні та блокові (рис. 5.6).



а) б) в) г)

Рисунок 5.6 – Структури БМК різних типів:
а) з горизонтальними та вертикальними каналами;
б) з вертикальними каналами;
в) безканальні; г) блокові

У канальних БМК (рис. 5.6, а) в центральній ділянці кристала розміщені БО (позначені квадратами) та канали трасування. У канальних БМК здатність створення зв'язків нівелюється малою площею упакування внаслідок значних затрат площі кристала на ділянці міжз'єднань. Для підви-

щення ефективності упакування БО канали організовуються тільки вертикально (див. рис. 5.6, б).

Пошук можливостей створення БМК високого рівня інтеграції привів до безканальної архітектури БМК. Внутрішня ділянка такого БМК має упаковані щільно ряди БО та не містить фіксованих каналів для трасування міжз'єднань (див. рис. 5.6, в). У кристалі цього типу будь-яка ділянка, в якій є БО, може застосовуватись для створення логічної схеми та міжз'єднань. Більш раціональне розташування зв'язків у безканальних БМК дозволяє зменшити затримку передачі сигналу по зв'язках і, як наслідок, зменшити довжину та паразитні ємності міжз'єднань.

Безканальні БМК створюються у варіантах «море вентилів» і «море транзисторів». У першому випадку кристал має масив закінчених логічних елементів, у другому – масив транзисторів. Для покращення ефективності створення на базі БМК функціонально складних пристроїв, що мають не лише комбінаційні схеми, але й велике число елементів пам'яті, в сучасні БМК високого ступеня інтеграції, крім БО, додають спеціалізовані блоки пам'яті, помножувачі, схеми прискореного перенесення тощо, створені на рівні топології кристала-заготовки. Такі БМК є блоковими (див. рис. 5.6, г).

БМК відносять до категорії напівзамовних ПЛІС. Тобто, масово виготовляються заготовки (напівфабрикати) НВІС, а надання кристалам-заготовкам індивідуального вигляду відбувається на завершальній стадії виробництва. Під час проектування цифрових пристроїв на БМК потрібно користуватися бібліотеками функціональних осередків, що містять готові схемні рішення, які створюються розробниками мікросхеми з урахуванням фізичних характеристик кристала.

Оцінення логічної складності БМК відбувається за допомогою еквівалентного вентиля – групи елементів БМК для створення логічної функції двовходового вентиля «І-НІ» чи «АБО-НІ».

5.1.5 Пристрої на основі програмованих користувачем вентильних матриць FPGA

Подальший розвиток архітектури БМК поданий у вигляді програмованих користувачем вентильних матриць (FPGA). Вони, на відміну від БМК, можуть програмуватися користувачем «на місці» без здійснення додаткових виробничих циклів.

Внутрішня організація FPGA подібна до канальних БМК (масив БО на внутрішній площі кристала та масив периферійних осередків по периметру, трасувальні канали для зв'язків між БО). Відмінність FPGA від БМК – в структурі цих компонентів кристала.

Розглянемо топологію кристала ПЛІС (FPGA) серії Virtex фірми Xilinx (рис. 5.7).

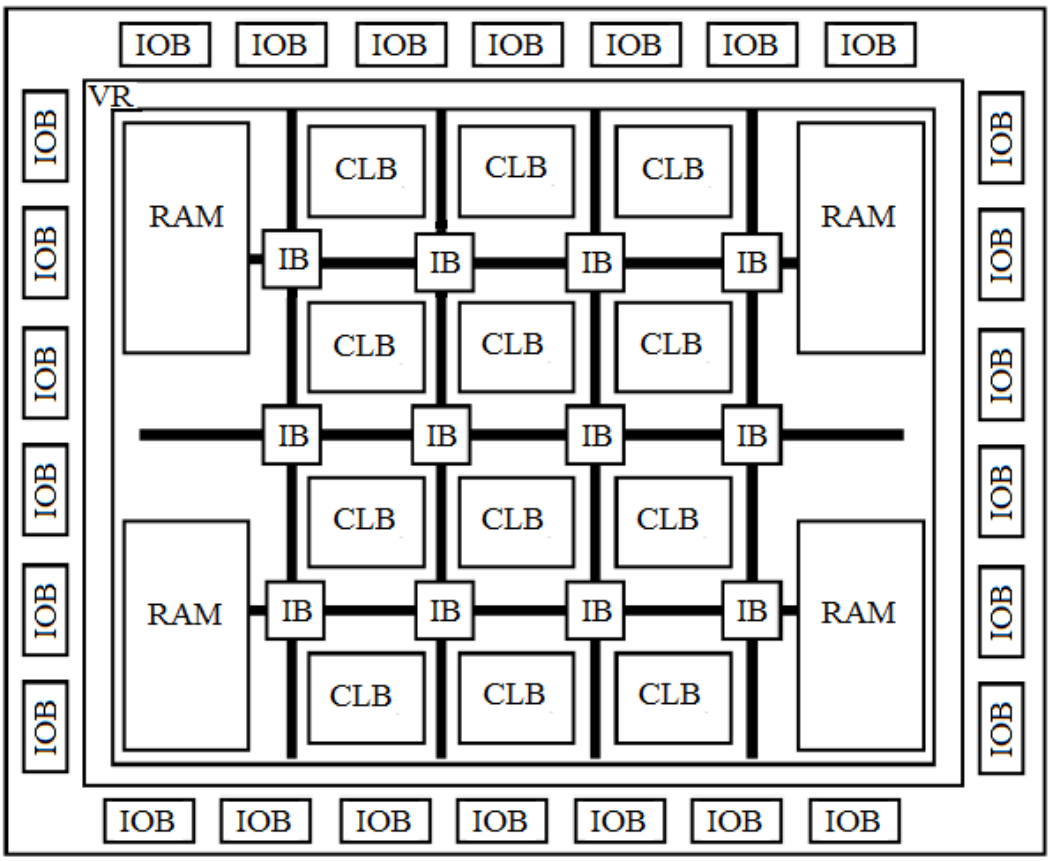
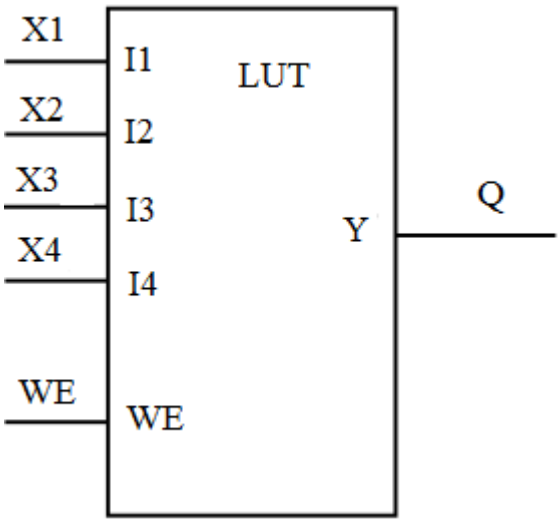


Рисунок 5.7 – Топологія кристала ПЛІС
серії Virtex фірми Xilinx

На площі кристала ПЛІС по периметру розташовані: матриця конфігурованих логічних блоків (КЛБ або CLB); матриця відрізків міжз'єднань, вкритих матрицями з польових транзисторів – перемичок IB; блоки налаштування RAM; блоки введення/виведення сигналів IOB та периферійний канал міжз'єднань з мінімальною затримкою VR. Забезпечення достатніх можливостей маршрутизації міжз'єднань і мінімальної затримки при передачі сигналів здійснюється за рахунок використання до дев'яти шарів металізації.

БО в архітектурі FPGA набуває вигляду КЛБ, основним логічним елементом якого є логічна таблиця (ЛТ або LUT), яка становить RAM і створена для зберігання логічної функції від n аргументів. Фактично в пам'ять заноситься таблиця істиності булевої функції, причому набір значень аргументів використовується як адреса блоку пам'яті, за яким на вихід приходить значення функції. Для задання таблиці істинності логічної функції від n аргументів потрібно використати однобітовий блок пам'яті, що містить 2n осередків. Позначення чотиривходового блоку ЛТ для задання функції «І» від 4 аргументів (16 осередків пам'яті) за таблицею істинності показано на рис. 5.8.

В склад КЛБ звичайно входять декілька ЛТ, а також комбінаційні та тригерні схеми (рис. 5.9).



а)

X4	X3	X2	X1	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

б)

Рисунок 5.8 – Блок LUT: а) позначення; б) таблиця істинності функції «І» від 4 аргументів

КЛБ мають такі властивості:

- функціональність визначає наскільки великі логічні можливості 1 КЛБ;

- зернистість визначає наскільки функціонально та схемотехнічно простими будуть складові елементи КЛБ.

Прикладом дрібнозернистого КЛБ може бути логічний блок фірми Crosspoint Solutions, який містить ланцюги транзисторів, а прикладом ве-

ликозернистого КЛБ – мікросхеми Xilinx XC4000, які мають 2 ЛТ на 4 аргументи, 1 ЛТ на 3 аргументи, 2 D-тригери, які зв’язані через декілька мультиплексорів, а також логічні схеми вентильного типу.

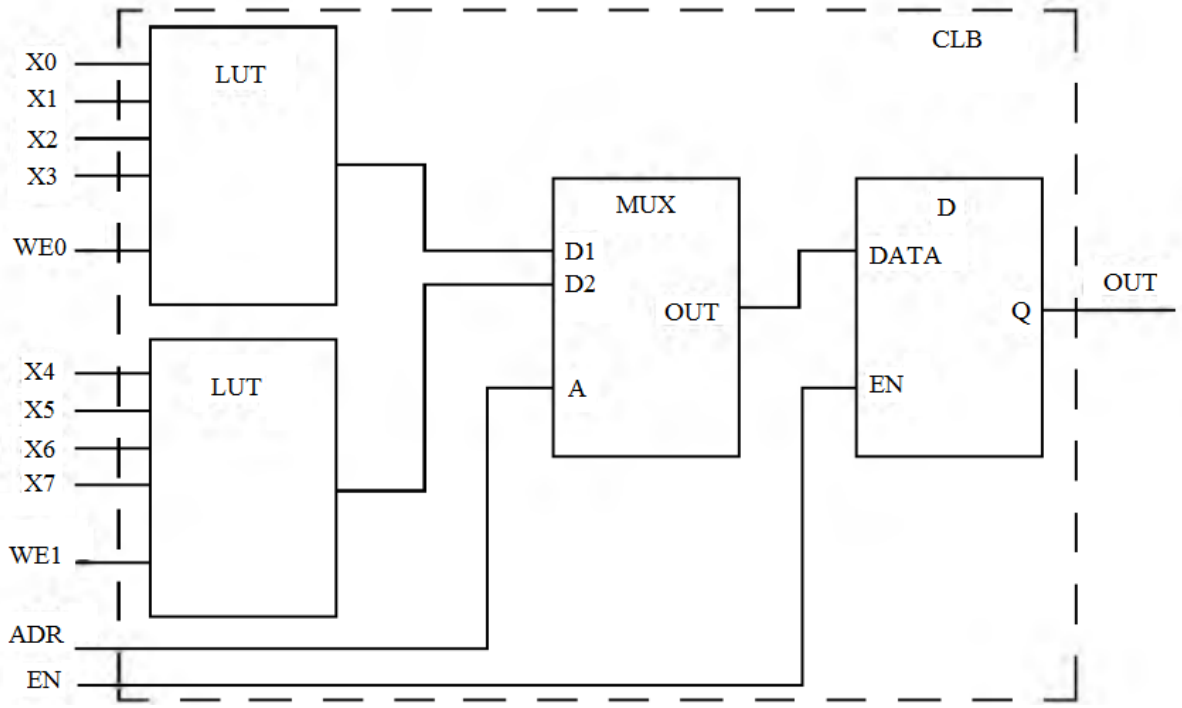


Рисунок 5.9 – Структурна схема КЛБ

Дрібнозернистість КЛБ дозволяє більш широко їх використовувати та реалізовувати відтворювані функції різними способами. Але дрібнозернистість ускладнює систему міжз’єднань FPGA через велике число програмованих точок зв’язку.

Мікросхеми FPGA характеризуються складною ієрархічною системою зв’язків, яка містить зв’язок загального призначення, довгі лінії для передавання на великі відстані з малою затримкою, прямі зв’язки та лінії тактування.

Універсальність і зручність налаштування КЛБ вимагають значних витрат і суттєво впливають на швидкодію пристроїв на FPGA. Підвищення технічних характеристик у цих пристроях відбувається за рахунок інтеграції у кристал готових функціональних блоків (блоків пам’яті, помножувачів, процесорних ядер). Подібну структуру мають мікросхеми FPGA Xilinx Virtex-II Pro.

Незважаючи на обмежені функціональні та швидкісні властивості, порівняно з напівзамовними ПЛІС, FPGA є досить перспективним середовищем для створення цифрових пристроїв. Важлива перевага FPGA полягає у простоті конфігурації мікросхеми. При вмиканні живлення сигнали конфігурації («прошивка») переписуються в спеціальний зсувний регістр ПЛІС, до виходів якого підключено затвори всіх програмованих транзисторів (рис. 5.10).

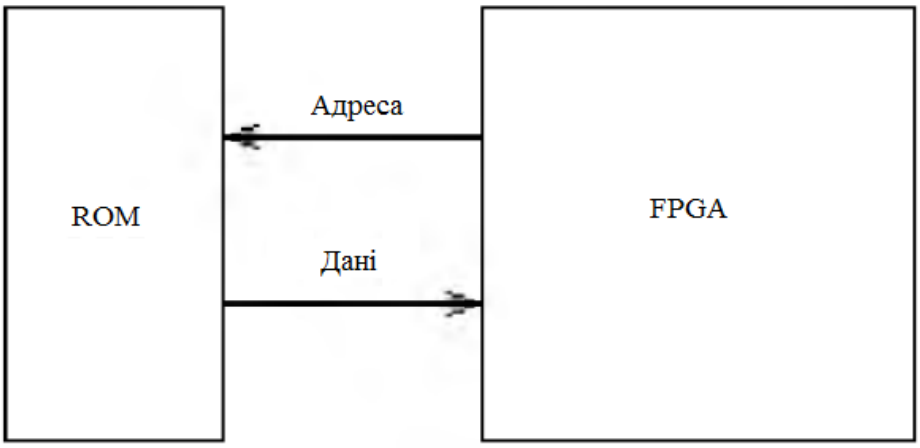


Рисунок 5.10 – Структурна схема цифрового пристрою на основі FPGA

Найбільш ефективно переваги мікросхем FPGA виявляються під час використання оптимізованих бібліотек компонентів від розробників ПЛІС з урахуванням топологічних властивостей кристала.

Порівняно з СПЛП, FPGA мають переваги, які пов'язані з необмеженою кількістю перепрограмувань, більшою питомою ємністю вентилів на 1 цент, низьким рівнем енергоспоживанням, більш високою надійністю.

5.2 Програмовані логічні інтегральні схеми

5.2.1 Розвиток архітектури програмованих логічних інтегральних схем

Розвиток архітектури ПЛІС йде в напрямку створення комбінованих структур, які об'єднують переваги FPGA та CPLD. Такі пристрої називаються програмованою логікою змішаної архітектури FLEX. Зберігши деякі характеристики, які присутні в CPLD, мікросхеми FLEX мають логічні елементи табличного типу LUT, їх логічні блоки організовані у вигляді матриці з горизонтальними та вертикальними трасувальними каналами, характерними для FPGA. Наявність безперервних зв'язків, що типово для CPLD, дає малу затримку сигналів. Представниками мікросхем цього типу є ПЛІС фірми ALTERA FLEX 8000, FLEX 10K.

Перші ПЛІС, які створювалися за технологією транзисторно-транзисторної логіки з діодами Шотткі (до середини 80-х років XX ст.), характеризувалися високою швидкодією (менше 10 нс), низькою вартістю, великою споживчою потужністю, малим ступенем інтеграції, неможливістю перепрограмування. В кінці 80-х років їх місце зайняли ПЛІС, створені на основі комплементарної структури метал-оксид-напівпровідник (КМОН). В них сполучними елементами (перемичками) є осередки пам'яті типу EPROM або EEPROM. Якщо в біполярних ПЛІС з'єднання порушується внаслідок звичайного записування перемички, то в КМОН ПЛІС осередку програмуються внаслідок накопичення чи видалення електричного заряду. Ці перемички можна не лише розривати, але й створювати. Цей

процес має назву стирання схеми. Залежно від типу елементів пам'яті існують ПЛІС з ультрафіолетовим (УФ) стиранням (EPROM) та електричним стиранням (EEPROM). ПЛІС з УФ стиранням виробляються в керамічних корпусах з віконцем. Стирання здійснюється під час опромінення ПЛІС УФ випромінюванням із попередньо заданими параметрами. Стирання ПЛІС типу EEPROM здійснюється внаслідок надходження на схему певних електричних сигналів (15–25 В). Впровадження технології КМОН дозволило значно збільшити ступінь інтеграції ПЛІС та досягти 10000 і більше вентилів.

Постійні технологічні вдосконалення та зменшення топологічних норм проектування дозволили отримати рівень інтеграції сучасної програмованої логіки на рівні кількох мільйонів еквівалентних вентилів при робочій тактовій частоті до 500 МГц. Завдяки цьому стало можливим розташувати на кристалі схему високої складності – систему, яка містить мікропроцесор, пам'ять, схеми сполучення тощо. Такі системи називаються системами на кристалі. При цьому різні функціональні блоки створюються одними й тими ж апаратними засобами. Системи різного призначення зазвичай мають типові компоненти, що вимагають введення поряд зі структурами програмованої логіки спеціальних областей з визначеними заздалегідь функціями – апаратних ядер (Hardcores). Введення спеціальних апаратних ядер у ПЛІС призводить до зменшення універсальності ПЛІС з Hardcores, порівняно з стандартними мікросхемами програмованої логіки.

5.2.2 Системний підхід до проектування пристроїв на програмованих логічних інтегральних схемах

На сьогодні проектування складних цифрових пристроїв, які містять сотні тисяч або навіть мільйони компонентів, вимагає використання системного підходу. Він ґрунтується на тому, що проектована система може бути подана декількома різними за формою та адекватністю моделями. Як цифрова система розглядається пристрій, який перетворює чи зберігає інформацію.

Системний підхід до проектування цифрових пристроїв передбачає проведення початкового опису всього пристрою як абстрактної структури, яка має задовольняти потрібні умови – специфікації. Далі ця структура (велика система) розділяється на підсистеми, кожна з яких виконує незалежну функцію, але разом ці підсистеми виконують функції великої системи. На наступному етапі кожна підсистема поділяється на складові нижчого рівня. Цей процес продовжується до тих пір, поки на найнижчому рівні модель пристрою не буде складатися тільки з найпростіших (неподільних) складових.

Перевагою цього є незалежність під час проектування кожної з підсистем. Також проектувальнику не потрібно працювати з усім обсягом інформації, а лише з тим, який потрібний для створення конкретної частини проекту.

Основні етапи системного проектування цифрових пристроїв наведено на рис. 5.11.

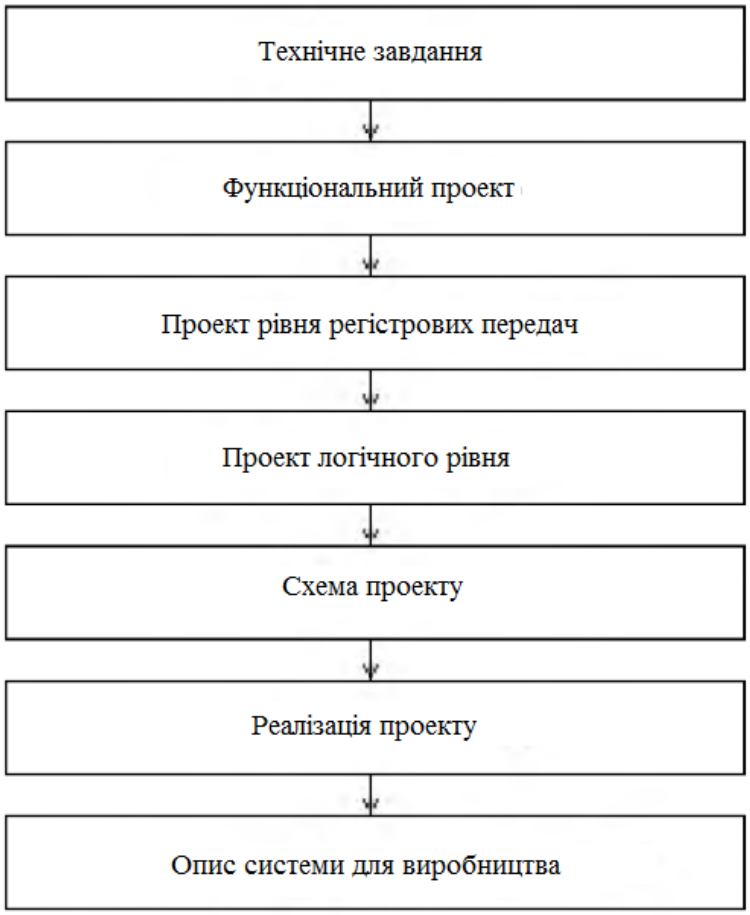


Рисунок 5.11 – Етапи системного проектування цифрових пристроїв

Перший етап проектування – створення технічного завдання (ТЗ), яке містить вимоги до робочих характеристик системи, опис інтерфейсу, експлуатаційні характеристики системи. Таким самим чином відбувається оцінювання ефективності системи, розрахунок вартості готового виробу тощо.

Другий етап проектування – створення на основі ТЗ попередньої високорівневої функціонально-структурної схеми системи з описом алгоритмічних основ роботи кожного компонента.

Третій етап проектування – перетворення отриманої функціонально-структурної схеми в модель рівня реєстрових передач, яка містить описи регістрів, модулів пам’яті, операційних автоматів і автоматів керування.

Наступний етап проектування передбачає створення логічних схем для кожного компонента. При цьому він розглядається у вигляді системи, що складається з певної кількості найпростіших логічних вентилів.

Далі опис логічного рівня переходе в схемний (створюється принципова схема пристрою на рівні транзисторів).

Останніми етапами є проектування топології кристала, обрахунок фізичних властивостей пристрою (площа кристала та розсіювана потужність). При цьому проходить верифікація проектних норм, встановлюються фізичні параметри схеми та готується повний опис пристрою для здійснення виробничих циклів.

Кожний етап проектування цифрового пристрою супроводжується різними (за повнотою опису та ступенем адекватності) моделями, які можна поділити на 3 типи:

1. *Функціональні* – описують особливості роботи пристрою без урахування його технічної реалізації. Формою подання пристрою на даному рівні є булеві рівняння, алгоритми, функції;

2. *Структурні* – вводять в опис пристрою структуру для реалізації певних функцій. Це дозволяє розширити реалізацію поведінкової моделі до рівня логічних вентилів і підвищити адекватність опису пристрою;

3. *Фізичні* – подають пристрої в конкретному схемотехнічному базисі, що дозволяє провести повну оптимізацію апаратної реалізації.

На всіх етапах системного проектування використовуються компоненти. На найвищому рівні – це невелика кількість складних компонентів (суматори чи модулі пам’яті), на нижніх – величезне число елементарних компонентів (логічні вентилі чи транзистори). Кожен рівень має свій математичний опис. Вивчення фізичних характеристик проекту та оптимізацію параметрів схеми потрібно розглядати на нижніх рівнях проектування, хоч це й вимагає довшого часу на розробку та аналіз.

5.2.3 Методика та способи проектування пристроїв на програмованих логічних інтегральних схемах

Створення пристроїв високої складності на ПЛІС здійснюється лише за допомогою спеціальних систем автоматизованого проектування (САПР). При цьому структурна схема етапів проектування (див. рис. 5.11) трансформується в алгоритм автоматизованого проектування ПЛІС, який наведений на рисунку 5.12.

На концептуальному рівні визначаються функції пристрою, вхідні й вихідні сигнали, можливості розподілу проекту на частини. Цей рівень проектування не пов’язаний з автоматизацією, його реалізація повністю залежить від розробника.

Результати концептуального синтезу вносяться в САПР, в якій відбувається компіляція проекту, тобто синтез пристрою в базисі бібліотеки моделей. Компіляція відбувається у вигляді послідовних етапів: формування бази даних проекту, контроль з’єднань, логічна мінімізація проекту тощо. Результатом компіляції є файл, який має конфігураційну інформацію для заданої ПЛІС.

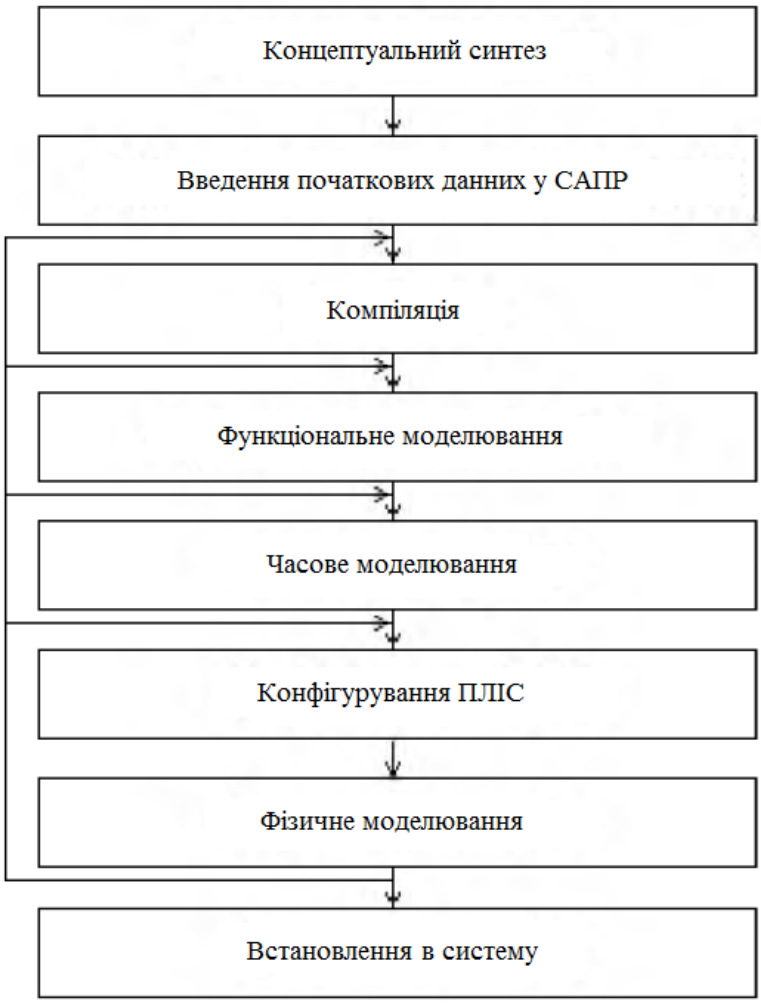


Рисунок 5.12 – Алгоритм автоматизованого проектування ПЛІС

Скомпільований проект ретельно перевіряється, тому за етапом синтезу слідує етап аналізу шляхом моделювання та теоретичної верифікації. Моделювання відбувається на кількох рівнях й характеризується різним ступенем подання властивостей реального об’єкта. Наприклад, за допомогою функціонального моделювання можна перевірити логічну структуру пристрою. Часове моделювання дає можливість проводити різноманітні тести роботи пристрою з урахуванням затримки сигналів у компонентах без урахування топології трасування.

В процесі моделювання можуть виникати помилки, що потребують усунення. Це надає процесу проектування ітеративності через повернення до попередніх етапів і введення потрібних змін в проект.

Потім відбувається конфігурація ПЛІС, після чого здійснюється перехід до фізичного моделювання, а саме; перевірки реальної роботи спроектованого пристрою. В разі успішного завершення фізичного моделювання пристрій стає готовим для встановлення в систему.

Використання САПР для створення пристроїв на ПЛІС вимагає ефективних, наочних, керованих і контрольованих способів опису проекту. На сьогодні найбільш поширеними є графічний і текстовий.

Графічний спосіб базується на поданні схеми проекту в базисі допустимих для даної САПР бібліотечних елементів. Переваги графічного способу полягають у традиційності та наочності. Однак в разі створення пристроїв на ПЛІС, які містять десятки чи сотні тисяч еквівалентних вентилів, різко втрачається наочність.

Тому набув істотного розвитку *текстовий опис* цифрових пристроїв на ПЛІС, реалізований у вигляді мов опису апаратури (HDL).

Сучасні мови опису апаратури поєднують опис пристрою, що проектується, з точки зору його поведінки (функцій, що виконуються) та його структури. Це дозволяє подати проект у формі текстового опису алгоритмів функціонування окремих компонентів пристрою разом з описом міжкомпонентних з'єднань.

Перевагами такого способу опису є компактність, автоматизація більшості перетворень, можливість перенести проект з однієї апаратної платформи на іншу, проста документація.

5.2.4 Характеристики новітніх програмованих логічних інтегральних схем

На сьогодні передовими компаніями, які займаються розробкою ПЛІС, є Intel, Xilinx та Achronix. Останніми їхніми розробками є ПЛІС серії Agilex (Intel), ПЛІС серії Versal ACAP (Xilinx), ПЛІС Speedster 7t (Achronix). Таким чином, на ринку є 3 добре диференційовані пропозиції ПЛІС високого класу, що виробляються за передовими технологіями та з унікальними особливостями й можливостями.

Останнім часом відбувається збільшення темпів поширення систем зв'язку на основі покоління 5G. Причому перші системи 5G базуються на ПЛІС попереднього покоління. ПЛІС нового покоління дадуть поштовх до поширення покоління 5G. Зараз з ймовірністю 99% сигнал будь-якого мобільного телефону проходить через декілька ПЛІС мережі стільникового зв'язку. В разі впровадження покоління 5G вплив ПЛІС буде ще помітнішим.

Аналогічно розширюється впровадження ПЛІС в системах штучного інтелекту, ринок яких постійно зростає. Кожен із вищевказаних розробників ПЛІС дає зрозуміти, що захоплення цього ринку задля прискорення впровадження штучного інтелекту є в пріоритеті, тому вони створили свої нові чіпи з підтримкою цієї технології. Комбінація зазначених факторів дозволила компаніям жорстко конкурувати одна з одною як в освоєнні зв'язку 5-го покоління, так і штучного інтелекту.

Таким чином, потреба у вирішенні задач штучного інтелекту, зокрема в нейронних мережах, штовхає компанії-розробники ПЛІС на збільшення обчислювальних ресурсів ПЛІС до рекордних значень. Апаратна підтримка арифметики з ПК стає обов'язковою. Надмірні обчислювальні ресурси ПЛІС сприяють суттєвому погіршенню результатів їх розміщення та трасування на кристалі. Вирішення цієї проблеми відбувається за рахунок мережі на кристалі (NoC) зі спрощенням звичайних засобів трасування

зв’язків ПЛІС. Це дозволяє здійснювати розміщення та трасування над невеликими кластерами, розташування яких на кристалі стає довільним. Така архітектура узгоджується з алгоритмами нейронних мереж, які вимагають, щоб штучні нейрони паралельно взаємодіяли з якомога більшою кількістю своїх сусідів. Надмірні обсяги ресурсів ПЛІС призводять до рекордних цін (десятки тисяч дол.), які через гальмування закону Мура не будуть суттєво зменшуватись і майбутньому. Ефективність застосування ресурсів ПЛІС та їх розміщення і трасування також залежить суттєво від інтелектуального рівня САПР, що покликані вирішити ці задачі, а також від самих алгоритмів, зокрема, від алгоритмів штучного інтелекту.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Дайте пояснення особливостям архітектури ПЛМ, а також поясніть принципи проектування цифрових пристроїв на їхній основі.
2. Дайте пояснення відмінностям архітектури ПЛМ і ПМЛ.
3. Які існують сучасні тенденції розвитку архітектур ПЛМ і ПМЛ?
4. Дайте пояснення принципам побудови пристроїв на БМК.
5. В чому полягає логічна складність пристроїв на БМК?
6. Дайте пояснення переваг і недоліків різних структур БМК.
7. Дайте пояснення принципам внутрішньої організації ПЛІС FPGA.
8. Дайте пояснення структури конфігураційного логічного блоку FPGA.
9. Яка функція логічної таблиці LUT в FPGA?
10. Дайте пояснення принципу конфігурації ПЛІС FPGA.
11. Які сучасні концепції розвитку архітектур ПЛІС?
12. Дайте пояснення системному підходу під час проектування цифрових пристроїв.
13. Які функціональні елементи утворюють систему на кристалі?
14. Перерахуйте етапи системного проектування цифрових пристроїв.
15. Як відбувається зміна адекватності формального опису цифрового пристрою під час переходу від функціональної до структурної та до фізичної моделей.
16. Які особливості автоматизованого проектування цифрових пристроїв на ПЛІС?
17. Дайте пояснення основним перевагам та недолікам способів схемного та текстового уявлень цифрових пристроїв на ПЛІС.
18. Дайте пояснення основним особливостям мов опису апаратури високого і низького рівнів.

ТЕСТ ДЛЯ САМОКОНТРОЛЮ

1. На основі якого керування працює АЛП?
   а – мікропрограмного;
   б – ручного;
   в – працює автономно;
   г – сигналів керування.
2. Які тип КА може використовуватись в АЛП?
   а – зі схемною логікою;
   б – з програмованою логікою;
   в – зі схемною та програмованою логіками;
   г – жоден.
3. За скільки тактів АЛП виконує додавання та віднімання?
   а – 3;
   б – 4;
   в – 5;
   г – 7.
4. При яких операціях спочатку проводится перевірка на 0?
   а – додавання та віднімання;
   б – додавання та ділення;
   в – множення та віднімання;
   г – множення та ділення.
5. Скільки режимів роботи має пристрій керування з жорсткою логікою?
   а – 1;
   б – 2;
   в – 3;
   г – 4.
6. Скільки режимів роботи має ґонтролер послідовності мікрокоманд?
   а – 1;
   б – 2;
   в – 3;
   г – 4.
7. Що і чого є першими представниками?
   а – ПЛІС є першими представниками ПЛМ;
   б – ПЛМ є першими представниками ПЛІС;
   в – ПМЛ є першими представниками ПЛМ;
   г – МЛБ є першими представниками ПЛІС.
8. Які на сьогодні компанії з розробки ПЛІС є провідними?
   а – Intel, Xilinx, ALTERA;
   б – AMD, Sony, Achronix;
   в – Intel, Panasonic, Achronix;
   г – Intel, Xilinx, Achronix.

СПИСОК ЛІТЕРАТУРИ

1. Білинський Й. Й., Гикавий В А., Мельничук А. О. Цифрова схемотехніка : навчальний посібник. Частина 1. Базові поняття цифрової схемотехніки. / Вінниця : ВНТУ, 2011. 133 с.

2. Білинський Й. Й., Ратушний П. М., Мельничук А. О. Цифрова схемотехніка : навчальний посібник. Частина 2. Електронні пристрої і системи. Вінниця : ВНТУ, 2017. 171 с.

3. Азаров О. Д., Гарнага В. А., Клятченко Я. М., Тарасенко В. П. Комп'ютерна схемотехніка : підручник. Вінниця : ВНТУ, 2018. 230 с.

4. Чураков А. Я., Шаров С. В., Строкань О. В. Архітектура ЕОМ : навчальний посібник. Мелітополь : Видавництво МДПУ ім. Богдана Хмельницького, 2014. 180 с.

5. Тарарака В. Д. Архітектура комп'ютерних систем : навчальний посібник. Житомир : ЖДТУ, 2018. 384 с.

6. Мілих В. І., Шавьолкін О. О. Електротехніка, електроніка та мікропроцесорна техніка : підручник. К. : Каравела, 2007. 688 с.

7. Мельник А. О. Архітектура комп'ютера. Луцьк : Волинська обласна друкарня, 2008. 470 с.

8. Коркішко Т., Мельник А., Мельник В. Алгоритми та процесори симетричного блокового шифрування. Львів : БаК, 2003. 168 с.

9. Угрюмов Е. П. Цифровая схемотехніка. СПб. : БХВ, 2000. 528 с.

10. Лехин С. М. Схемотехника ЭВМ . СПб. : БХВ, 2010. 672 с.

11. Харрис Д. М., Харрис С. Л. Цифровая схемотехника и архитектура комп'ютера. Волтем : Morgan Kaufmann Publishers, 2013. 1662 с.

12. Комп'ютерна схемотехніка : навчальний посібник / Азаров О. Д. та ін. Вінниця : ВНТУ, 2015. 134 с.

Навчальне видання

Білинський Йосип Йосипович
Книш Богдан Петрович

ЦИФРОВА СХЕМОТЕХНІКА
Електронно-обчислювальні пристрої

Навчальний посібник

Рукопис оформив *Б. Книш*

Редактор *В. Дружиніна*

Оригінал-макет підготувала *Т. Криклива*

Підписано до друку 22.09.2021.
Формат 29,7×42 ¼. Папір офсетний.
Гарнітура Times New Roman.
Друк різографічний. Ум. друк. арк. 3,96.
Наклад 50 (1-й запуск 1-21) пр. Зам. № 2021-098.

Видавець та виготовлювач
Вінницький національний технічний університет,
інформаційний редакційно-видавничий центр.
ВНТУ, ГНК, к. 114.
Хмельницьке шосе, 95,
м. Вінниця, 21021.
Тел. (0432) 65-18-06.
press.vntu.edu.ua;
E-mail: kivc.vntu@gmail.com
Свідоцтво суб'єкта видавничої справи
серія ДК № 3516 від 01.07.2009 р.