МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

# МЕТОДИЧНІ ВКАЗІВКИ

**до лабораторного практикуму з курсу "Архітектура комп'ютерів" для студентів спеціальностей "Комп'ютерні системи і мережі" і "Системне програмування" денної і заочної форм навчання**

**Донецьк ДНТУ 2005**

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

# МЕТОДИЧНІ ВКАЗІВКИ

до лабораторного практикуму з курсу "Архітектура комп'ютерів" для студентів спеціальностей "Комп'ютерні системи і мережі" і "Системне програмування" денної і заочної форм навчання

Затверджено
на засіданні кафедри
Електронних
обчислювальних машин.
Протокол № 2 від
27. 09. 2004 р.

Затверджено
на засіданні учбово-видавничої
ради ДНТУ.
Протокол № 15 від 01. 12. 2004 р.

Донецьк ДНТУ 2005

УДК 681.3

Методичні вказівки до лабораторного практикуму з курсу "Архітектура комп'ютерів" для підготовки фахівців і магістрів на базі напрямку "Комп'ютерна інженерія" / Укл. В. В. Лапко, Ю.В. Губарь. – Донецьк: Видавництво ДНТУ, 2005. - 120 с.

Збірник містить методичні вказівки з 12 лабораторних робіт. У збірник увійшли лабораторні роботи з проектування, аналізу і симуляції операційних та керуючих пристроїв, виконаних на базі мікропроцесорного комплекта з розрядно – модульною організацією на основі мікросхем серії К1804 (закордонний аналог мікросхеми серії Am2900 фірми Advanced Micro Devices, США). Розглянуті питання мікропрограмування та розробки системи синхронізації блоків множення, ділення, підсумовування та віднімання з фіксованою та рухомою комою, виконання симуляції і тестування роботи мікропроцесорних систем на функціональному рівні з використанням програмного пакету Active – HDL фірми ALDEC (США).

Укладачі — доц. Лапко В.В., доц. Губарь Ю.В.

Відповідальний за випуск — проф. Святний В.А.

Рецензент — проф. Аверін Г. В.

Автори висловлюють подяку студентам Шепілю О. В. (гр.СП-00н), Доста М. О. (гр.СП-00н) та Цололо С. О. (гр.ВТ-00а) за сприяння у виданні методичних вказівок до лабораторного практикуму.

## ПЕРЕЛІК ОСНОВНИХ СКОРОЧЕНЬ

**HDL (Hardware Description Language)** – мова опису апаратних засобів.

**VHDL (Very High – Speed Integrated Circuit Hardware Description Language)** – мова опису апаратних засобів в рамках проекта VHSIC (надшвидкісних інтегральних схем).

**ROM (Read Only Memory)** – постійний запам'ятовуючий пристрій.

**RAM (Random Access Memory)** – оперативний запам'ятовуючий пристрій.

**МОП** – мікропроцесорний обчислювальний пристрій.

**БОД** – блок обробки даних.

**МПС** – мікропроцесорна секція.

**ВС1 і ВС2** - мікропроцесорні секції обробки даних К1804ВС1 та К1804ВС2.

**СПП** – схема прискорення переносу.

**СКАМ** – схема керування адресою мікрокоманд.

**ВУ1 і ВУ4** - мікропроцесорні секції формування адреси мікрокоманд.

**РМК** - регістр мікрокоманд.

**МПП** - мікропрограмна пам'ять.

**МХ** - мультиплексор.

**ЛАМК** - лічильник адреси мікрокоманд.

**РАМК** - регістр адреси мікрокоманд.

**ГСМ** - граф – схема мікропрограми.

**АЛП** - арифметично – логічний пристрій.

**РЗП** - регістровий запам'ятовуючий пристрій.

**БП** - буферний підсилювач.

**РК** - регістр команд.

**БС** - блок синхронізації.

**КП** - комбінаційний перетворювач.

# ВСТУП

Збірник містить методичні вказівки до циклу лабораторних робіт по ЕОМ з використанням мови апаратного опису VHDL та інструментального середовища Active – HDL фірми Aldec (США) [6 – 18]. У збірник увійшли лабораторні роботи з дослідження мікропроцесорних комплектів (МПК) серії К1804 (закордонний аналог Am2900, США), виконаних за біполярною технологією ТТЛ з діодами Шотткі [1, 2]. Висока швидкодія, мікропрограмне керування та розрядна – модульна організація мікросхем процесорних секцій дозволяють проектувати на їх основі обчислювальні пристрої широкого призначення.

Істотним недоліком цього МПК є складність верифікації працездатності спроектованого блоку на функціональному рівні в зв'язку з необхідністю перевірки виконання за тактами багатьох мікрокоманд мікропрограми. З використанням мови опису апаратури VHDL і системи Active – HDL з'явилася можливість на основі імітаційних VHDL – моделей спроектованого блоку автоматизувати тестування на функціональному рівні.

Для полегшення процесу проектування мікропроцесорних пристроїв є також доцільним побудувати бібліотеку VHDL – моделей (К1804_lib) основних компонентів МПК К1804 (ВС1, ВС2, ВУ1 і ВУ4). У цьому випадку побудова моделі пристрою зведиться до з'єднання готових до вживання модулів із складу бібліотеки та розробки користувачем самостійно деяких моделей схем обрамлення (таких як регістри, лічильники та інші), і підключення їх до реалізованого проекту.

Використання системи проектування Active – HDL , якою обладнані РС у навчальних лабораторіях кафедри ЕОМ з дозволу фірми Aldec, також сприяє полегшенню і прискоренню термінів створення, налагодження та аналізу результатів симуляції VHDL – проектів і сприяє підвищенню якості підготовки фахівців та магістрів в галузі комп'ютерної інженерії.

Збірник містить крім лабораторних робіт також додаток, в якому знаходиться короткий опис базових елементів МПК К1804.

**Лабораторна робота №1**

# ВИКОНАННЯ АРИФМЕТИЧНИХ ТА ЛОГІЧНИХ ОПЕРАЦІЙ З ВИКОРИСТАННЯМ МІКРОПРОЦЕСОРНОЇ СЕКЦІЇ К1804ВС1

**МЕТА РОБОТИ:** опанувати принципи функціонування мікропроцесорної секції (МПС) ВС1; придбати практичні навички реалізації найпростіших функцій; навчитися розробляти імітаційні моделі мовою VHDL вузли, блоки та мікроалгоритми реалізації конкретних задач; опанувати симуляцію роботи моделі пристроїв і аналіз отриманих результатів.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

Мікропроцесори серії К1804 належать до класу мікропрограмувальних ВІС з розрядно – модульною організацією. Побудова блоків обробки даних (БОД) необхідної розрядності виконується шляхом з'єднання необхідної кількості секцій ВС1, чим забезпечується велика гнучкість проектування пристроїв.

Для апробації архітектурних рішень та находження мікроалгоритмів роботи БОД на основі комплектів ВІС К1804 застосуємо описи пристрою та алгоритмів VHDL – мовою та симуляцію його роботи в часовому просторі з використанням VHDL – середовища.

Створення VHDL – проекту виконується в інструментальному середовищі проектування **Active – HDL** [ 7 ]. Вона включає різні програми для введення проектів, VHDL та Verilog компілятори, ядро моделювання, програмні модулі налагодження, графічний та текстовий висновки результатів симуляції. Привсдемо короткий опис необхідних для виконання роботи компонентів Active – HDL:

- **HDL Editor** - текстовий редактор, використовуваємий для створення вихідних файлів VHDL. Він інтегрований з моделюючим пристроєм, що полегшує налагодження вихідного текстового файлу.
- **State Diagram Editor** - редактор автоматів, призначений для редагування графів кінцевих автоматів.
- **Block Diagram Editor** - редактор блок – схем діаграм, призначений для створення моделей функціональних схем цифрових пристроїв (дозволяє вводити їх у вигляді рисунка). Редактор робить автоматичне перетворення графічного представлення проекту в код мови VHDL опису апаратури.
- **Console (Alt – 4)** - інтерактивне вікно для введення макрокоманд та виводу повідомлень з інструментального середовища.
- **Design Browser** - браузер проекту, призначений для відображення поточного стану проекту (файлів ресурсів проекту, складу робочих бібліотек, структури модуля проекту) та виконання процесу симуляції пристрою.
- **Waveform Editor** - редактор тимчасових діаграм, призначений для відображення результатів моделювання у формі часової діаграми. Редактор дозволяє редагувати форму сигналів для створення необхідних тестових векторів.
- **List** - вікно, у якому представлені результати моделювання обраних сигналів у виді текстової таблиці.
- **Watch (Alt – 5)** - інструмент налагодження, призначений для видачі поточних значень сигналів та перемінних під час моделювання.
- **Design Flow Manager** - керуюча програма, яка дозволяє автоматизувати процес проектування.

### 1.1 Послідовність виконання лабораторної роботи

1. Вивчити з використанням літературних джерел [1 – 5] і методичних вказівок склад, призначення і мікропрограмування МПС ВС1 та БОД на її основі.
2. Розробити граф – схему мікропрограми (ГСМ) обчислення заданої функції з використанням додаткових кодів. При цьому передбачити уведення у регістрову пам'ять ВС1 операндів з використанням зовнішньої шини D і зчитування отриманого результату на шину Y.
3. Розробити таблицю кодування (ТК), яка забезпечує обчислення заданої функції.
4*. Розробити програмну модель МПС ВС1.
5*. Розробити VHDL - модель 16 –го БОД з використанням і без використання схеми прискореного переносу (СПП) на основі мікросхеми К1804ВР1 [1].
6*. Виконати налагодження VHDL - моделі БОД і схем його обрамлення (регістри, мультиплексори та інші схеми).Провести симуляцію роботи БОД з використанням ТК.
7. Скласти звіт про виконану лабораторну роботу і захистити його.
8. Дати відповіді на контрольні запитання.

**Примітка.** Пункти 4*, 5* та 6* виконуються факультативно.

### 1.2 Варіанти індивідуальних завдань

Варіанти індівідуальних завдать наведено у табл. 1.2, де **N** – номер завдання.

Варіанти розподілу внутрішної пам'яті. Таблиця 1.1

| (N) m4 | X1 | X2 | X3 | X4 |
|---|---|---|---|---|
| 0 | R0 | RQ | R5 | R10 |
| 1 | R1 | R4 | R7 | RQ |
| 2 | R11 | R8 | R15 | R9 |
| 3 | R15 | R12 | RQ | R6 |

Варіанти індивідуальних завдань. Таблиця 1.2

| (N) m8 | Варіанти функцій |
|---|---|
| 0 | $F1 = 16(2X1+3X2-1) \oplus (X3-2X4)/8$ |
| 1 | $F2 = 8(3X1+4X2)+(3X3-1-2X4)/16$ |
| 2 | $F3 = 16[(2X1-1)-3X2] \wedge (X3+3X4)/4$ |
| 3 | $F4 = 8(2X1-3X2) \oplus (2X3+4X4-1)/16$ |
| 4 | $F5 = 2X1+4X2-1+X3/2+X4/8$ |
| 5 | $F6 = \overline{4(2X1+3X2)} \vee (X3/4-X4/2)$ |
| 6 | $F7 = \overline{8(2X1+X2/2)} \wedge (X3+3X4)/8$ |
| 7 | $F8 = [4(X1+2X2)] \oplus [(2X3-X4)/8]$ |

**Примітки.** Операнди X1 – X4 можуть вміщувати (4*K) біт, наприклад 16 біт і зображені у додатковому коду (старший розряд – знаковий) та надходять з зовнішньої шини D ВС1 у регістри внутрішньої пам'яті МПС (зазначені у табл. 1.1). Кількіть розрядів БОД розраховується з урахуванням необхідних операцій (табл. 1.2) та значення операндів (табл. 1.3)

Значення операндів для прикладу. Таблиця 1.3

| (N) m5 | X1 | X2 | X3 | X4 |
|---|---|---|---|---|
| 0 | -7536 | +12814 | +1718 | -365 |
| 1 | -12038 | +20356 | -108 | +1567 |
| 2 | +736 | -12445 | -1348 | +1408 |
| 3 | +1806 | -30556 | +2334 | -11035 |
| 4 | -9065 | +1056 | +31865 | -21775 |

### 1.3 Приклад виконання лабораторної роботи

Припустимо, що потрібно реалізувати з використанням БОД на основі МПС ВС1 функцію:

$$Y = 3 \cdot A + B - 2 \cdot C, \qquad (1.1)$$

де операнди **A, B** та **C** задані в додатковому коді. Очевидно, можливі наступні обчислювальні алгоритми реалізації функції (1.1):

| ***Алгоритм №1*** | ***Алгоритм №2*** | ***Алгоритм №3*** | ***Алгоритм №4*** |
|---|---|---|---|
| **1. Y := 0** | **1. Y := 0** | **1. Y := 0** | **1. Y := 0** |
| **2. Y := Y + A** | **2. Y := Y + A** | **2. Y := Y + A** | **2. Y := Y + A** |
| **3. Y := Y + A** | **3. Y := Y – C** | **3. Y := (Y – C)·2** | **3. Y := (Y + B) / 2** |
| **4. Y := Y + A** | **4. Y := Y + A** | **4. Y := Y + A** | **4. Y := Y + A** |
| **5. Y := Y + B** | **5. Y := Y – C** | **5. Y := Y + B** | **5. Y := (Y – C) · 2** |
| **6. Y := Y – C** | **6. Y := Y + A** | | |
| **7. Y := Y – C** | **7. Y := Y + B** | | |

Недоліком першого алгоритму є велика кількість необхідних розрядів БОД для уникнення переповнення при виконанні операцій 2 – 4 і 6 – 7.Більш кращим є другий алгоритм, у якому операції підсумовування та віднімання чергуються між собою. В третьому і четвертому алгоритмах використані властивості МПС ВС1, яка дозволяє в одному такті одночасно виконувати арифметичну операцію і операцію зсуву.

Розглянемо реалізацію функції Y за алгоритмом №4. Припустимо, що розподіл регістрової пам'яті має вигляд:

$$(A = 3) \rightarrow R1, \quad (B = 2) \rightarrow R2, \quad (C = 1) \rightarrow R3, \quad Y \rightarrow R0.$$

При цьому граф – схема мікропрограми (ГСМ) функції (1.1) має вигляд, який наведений на рис. 1.1. У ГСМ передбачається, що операнди перед обчисленням функції записуються у регістри ВС1 (вершини 0 – 2). З точки зору часу кожна операторна вершина

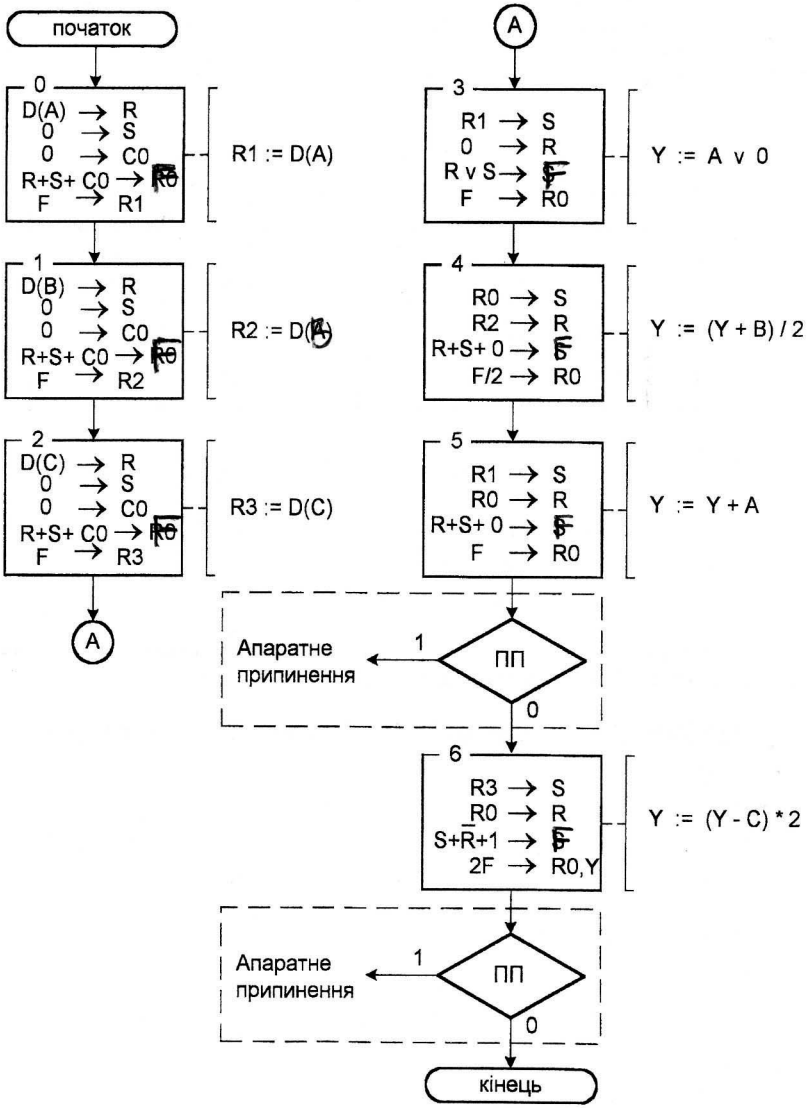



Рисунок 1.1 ГСМ реалізації функції Y=3·A + B – 2·C

ГСМ відповідає одній мікрокоманді (МК), яка може бути виконана в БОД за один такт дії сигналу синхронізації CLK.

При виконанні МК за номерами 5 і 6 будемо враховувати, що в схемі може виникнути сигнал переповнення (ПП = 1) на виводі БОД, який надходить на схему апаратного переривання і викликає блокування подачі тактових сигналів синхронізації і зупинку процесору.

Розглянемо приклад обчислення функції Y (для простості розглядається одна МПС):

$$\mathbf{Y = 2((A + B) / 2 + A - C) = 2\cdot[(0010 + 0011)/2 + 0011 - 0001] =}$$

$$\mathbf{= 2[0010.\underset{\text{PF0}}{\boxed{1}} + 0010] = 2[0100,\ PF0] = 1001.} \qquad (1.2)$$

З цього прикладу випливає, що Y може бути реалізована на одній секції, якщо для одержання коректного результату (**Y = 9**) необхідно зберігати значення спадаючого розряду PF0. Для цієї мети може бути використаний регістр -акумулятор PQ і операція зсуву подвійної довжини (див. табл. Д1.4 додатка).

При **I8 I7 I6 = 100 → R1 [ PF0 → PQ3 ];** якщо **I8 I7 I6 = 110 → L1 [ PQ3 → PF0 ].**

Слід зазначити, що при виконанні МК з номером 4 в одному такті виконуються арифметична операція (АО) ( **F : = A + B** ) та операція зсуву (ОЗ) праворуч ( **Y : = F / 2**). Взагалі при виконанні АО може виникнути ознака переповнення ПП = 1, яка зіпсує знак на виводі F АЛП. Для відновлення знаку потрібно при виконанні АО на вході PF3 формувати сигнал **F3 ⊕ ПП**. У цьому випадку сигнал переповнення зникне і перейде в нульовий стан ( **ПП = 0** ).

З обліком відзначеного вище, функціональна схема БОД зі схемами обрамлення має вигляд, який наведено на рис. 1.2. Мікропрограмна пам'ять (МПП) має вісім осередок (комірок) з 35 розрядами кожна. Вміст МПП (мікрокоманди) відбиває таблиця кодування ( табл. 1.4).

На основі таблиці кодування 1.4 треба підготувати у любому текстовому редакторі файл (наприклад, lab1_BC1.txt) формату, наведеному на рис. 1.3. Кожна мікрокоманда кодується в наступній послідовності: код інструкції BC1 I(8 – 0) розбивається на три поля (I86, I53, I20), у які пробіл або знак табуляції розділяють двійкові коди виконуваних операцій BC1; далі випливають поля адреси портів AA та AB у вигляді символів 16-ричного формату; за ними пробіл відділяє значення вхідного переносу C0 та сигналу OE (як 0 або 1); наприкінці файлу задається значення шини даних на 16 розрядів D у вигляді чотирьох 16 – ричних цифр.

Вхідний текстовий файл доцільно помістити в каталог проекту та користатися відносними шляхами при його читанні. Наприклад, "$dsn \ src \ lab1_BC1.txt", де $dsn - каталог, у якому знаходиться проект.

Перед початком симуляції необхідно виконати процес ініціалізації текстового файлу. Для цього з використанням стандартних функцій роботи з файлами здійснюється процес зчитування в МПП, аналіз та (при необхідності) переклад деяких полів мікрокоманди текстового файлу в двійковий код.

Для опису структури МПП доцільно використовувати пакет (***package***) - це засіб для виділення з програм та підпрограм загальних типів даних та функцій, що дозволяє спростити процес їхньої заміни [7, 9]. В розглядаємому випадку опис МПП має вигляд:

```
Package RomPack is
 file ROM_FILE: TEXT open read_mode is "$dsn \ src \ lab1_BC1.txt";
 constant WAIT_TIME: time : = 70 ns; -- затримка роботи МПП
 type ROM_TYPE is record              -- структура пам'яті
      I : bit_VECTOR ( 8 downto 0 );  -- інструкція BC1
```

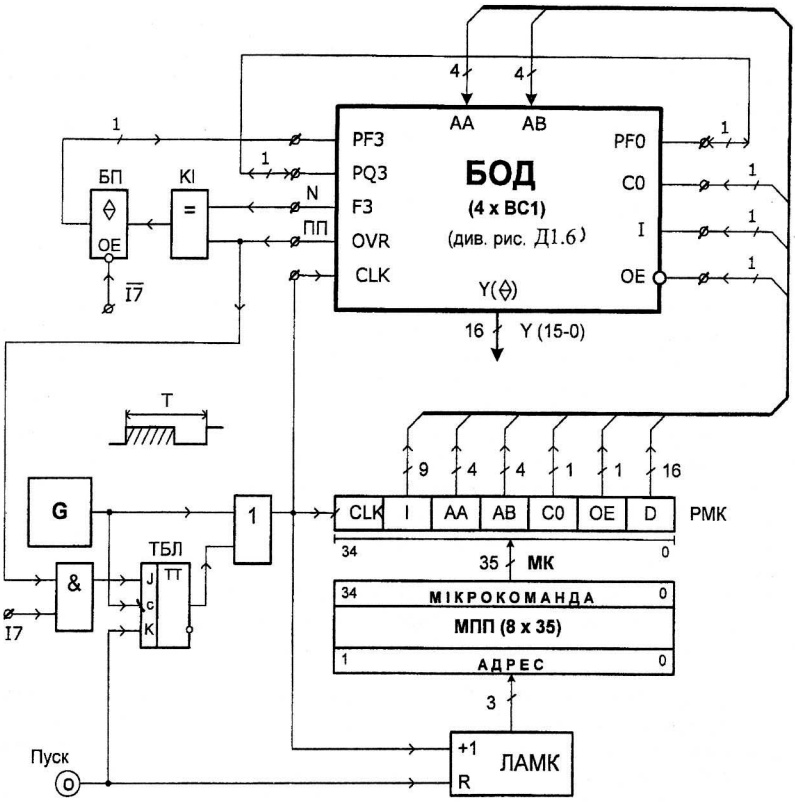



Рисунок 1.2. Функціональна схема БОД з обладнанням

З надходженням сигналу "Пуск" тригер блокування ( ТБЛ ) та лічильник адреси мікрокоманд ( ЛАМК ) скидаються в нульовий стан. Вміст ЛАМК при цьому вказує на адресу першої МК МПП. Вибрана з пам'яті мікрокоманда записується сигналом CLK (формується генератором G) у регістр мікрокоманд (РМК) і далі подається на керуючі входи БОД. Після виконання мікрокоманди на виводі БОД формуються ознаки результату. Керуючий інвертор ( KI) і буферний підсилювач з трьома станами (БП) формують сигнал на вході PF3 у відповідності з (1.3). У випадку виникнення сигналу ПП (при відсутності операції зсуву праворуч, тобто при I7 = 1) тригер встановлюється в одиничний стан і блокує подальше надходження тактових сигналів у ЛАМК, РМК та БОД.

$$PF3 = N \oplus ПП. \qquad (1.3)$$

**Таблиця кодування ГСМ** **Таблиця 1.4**

| № МК | МІКРОКОМАНДА | | | | | | | | Примітка | Результат операції (F) |
|---|---|---|---|---|---|---|---|---|---|---|
| | I8-I6 | I5-I3 | I2-I0 | AA | AB | C0 | $\overline{OE}$ | D | | |
| 0 | 3 | 0 | 7 | 0 | 1 | 0 | 0 | 1025 | D (A) → R1 | 0.001 0000 0010 0101 |
| 1 | 3 | 0 | 7 | 0 | 2 | 0 | 0 | 2237 | D (B) → R2 | 0.010 0010 0011 0111 |
| 2 | 3 | 0 | 7 | 0 | 3 | 0 | 0 | 2952 | D (C) → R3 | 0.010 1001 0101 0010 |
| 3 | 3 | 3 | 4 | 1 | 0 | * | * | * | R1 v 0 → R0 | 0.001 0000 0010 0101 |
| 4 | 4 | 0 | 1 | 2 | 0 | 0 | * | * | (R0 + R2)/2 → R0 | 0.001 1001 0010 1110 |
| 5 | 3 | 0 | 1 | 1 | 0 | 0 | * | * | R0 + R1 → R0 | 0.010 1001 0101 0011 |
| 6 | 6 | 1 | 1 | 3 | 0 | 1 | 0 | * | (R0 – R3)*2→R0,Y | 0.010 1110 0011 1110 |

11

| I86 | I53 | I20 | AA | AB | CO | $\overline{OE}$ | D |
|---|---|---|---|---|---|---|---|
| 011 | 000 | 111 | 0 | 1 | 0 | 0 | 1025 |
| 011 | 000 | 111 | 0 | 2 | 0 | 0 | 2237 |
| 011 | 000 | 111 | 0 | 3 | 0 | 0 | 2952 |
| 011 | 011 | 100 | 1 | 0 | 0 | 0 | 0000 |
| 100 | 000 | 001 | 2 | 0 | 0 | 0 | 0000 |
| 011 | 000 | 001 | 1 | 0 | 0 | 0 | 0000 |
| 110 | 001 | 001 | 3 | 0 | 1 | 0 | 0000 |

**Рисунок 1.3 Вхідний текстовий файл lab1_BC1.txt**

```
        AA : bit_VECTOR ( 3 downto 0 ) ;  -- адрес порта A
        AB : bit_VECTOR ( 3 downto 0 ) ;  -- адрес порта B
        D  : bit_VECTOR ( 15 downto 0 ) ; -- шина даних
        C0 : bit ; -- вхідной перенос
        OE : bit ; -- флаг отримання результату
    end record ;
    type ROMarr is array ( NATURAL range <> ) of ROM_TYPE ;
end RomPack ;
```

Тип МПП (***ROMarr***) визначений у VHDL -пакеті як масив значень типу ***ROM_ TYPE*** - тільки для операції зчитування. Тут же задається час затримки спрацьовування МПП (константа ***WAIT_TIME***), а також ім'я текстового файлу з указівкою режиму його відкриття та читання.

У БОД використовуються чотири секції ВС1, які вмикаються послідовно без використання СПП. Модель МПС ВС1 доцільно поместити в бібліотеку **K1804.lib**. Для її використання необхідно підключити цю бібліотеку до реалізованого проекту.

Відповідно до функціональної схеми 16-го БОД (див. рис. Д1.6) треба описати інтерфейс даного блоку (***entity*** **BOD** ***is***) та компоненти процесорної секції ВС1 (***component*** **BC1** ***is***) бібліотеки К1804. Далі описуються зв'язки сигналів низки переносів та коло зсуву чотирьох секцій ВС1. Ті клеми секцій ВС1, які не використовуються, можливо залишити відкритими (***open***). Наприклад, сигнал переповнення OVR використовується тільки в старшої МПС; в інших секціях ці клеми повинні залишатися відкритими.

Код інструкції ВС1 ( I ), адреси портів (AA та AB) подаються на всі секції. Код шини даних D розділяється між секціями по чотири біти на кожну секцію. Так як в даному проекті не потрібно виконувати аналіз результату АЛП на нуль, клеми Z всіх секцій також можуть бути залишені відкритими.

В розглядаємому випадку VHDL – опис БОД має вигляд:

```
entity BOD is
  port (
        DY : out std_logic_VECTOR ( 15 downto 0 ) ;
        CLK : in bit ;
        OE : in bit ;
        PF0 : inout std_logic ;
        PQ3 :  inout std_logic ;
        PF3 : inout std_logic ;
        PQ0 : inout std_logic ;
        C0  : in bit ;
        I : bit_VECTOR ( 8 downto 0 ) ;    -- інструкція BC1
        AA : bit_VECTOR ( 3 downto 0 ) ;  -- адрес порта A
        AB : bit_VECTOR ( 3 downto 0 ) ;  -- адрес порта B
        D  : bit_VECTOR ( 15 downto 0 ) ; -- шина даних
        OVR : out bit ;
        F3 : out bit ;
        ) ;
end BOD ;

architecture BOD of BOD is
component BC1 is
   port (
        PF0 : inout std_logic ;
        PQ3 : inout std_logic ;
        PF3 : inout std_logic ;
        PQ0 : inout std_logic ;
        CLK : in bit ;
        C0  : in bit ;
        I : bit_VECTOR ( 8 downto 0 ) ;    -- інструкція BC1
```

```
        AA : bit_VECTOR ( 3 downto 0 ) ;  -- адрес порта A
        AB : bit_VECTOR ( 3 downto 0 ) ;  -- адрес порта B
        D  : bit_VECTOR ( 15 downto 0 ) ; -- шина даних
        OVR : out bit ;
        F3 : out bit ;
        Z : out bit ;
        C4 : out bit ;
        P : out bit ;
        G : out bit ;
        Y out std_logic_VECTOR ( 3 downto 0 ) ;
        ) ;
end component ;

signal PF3_0, PQ3_0, PQ3_1, PF3_1, PF3_2, PQ3_2 : std_logic ;
signal C4_0, C4_1, C4_2, C4_3 : bit ;
begin
                --перша секція BC1
    MPS0 : BC1 port map ( PQ0 => open, PF0 => PF0, PQ3 => PQ3_0, PF3 => PF3_0, CLK => CLK, OE
=> OE, C0 => C0, I => I, AB => AB , AA => AA, D => D ( 3 downto 0 ), OVR => open, F3 => open, Z =>
open, C4 => C4_0, P => open, G => open, Y => DY ( 3 downto 0 ) );

                --друга секція BC1
    MPS1 : BC1 port map ( PQ0 => PQ3_0, PF0 => PF3_0, PQ3 => PQ3_1, PF3 => PF3_1, CLK => CLK,
OE => OE, C0 => C4_0, I => I, AB => AB , AA => AA, D => D ( 7 downto 4 ), OVR => open, F3 => open, Z
=> open, C4 => C4_1, P => open, G => open, Y => DY ( 7 downto 4 ) );

                --третя секція BC1
    MPS2 : BC1 port map ( PQ0 => PQ3_1, PF0 => PF3_1, PQ3 => PQ3_2, PF3 => PF3_2, CLK => CLK,
OE => OE, C0 => C4_1, I => I, AB => AB , AA => AA, D => D ( 11 downto 8 ), OVR => open, F3 => open, Z
=> open, C4 => C4_2, P => open, G => open, Y => DY ( 11 downto 8 ) );

                --четверта секція BC1
    MPS3 : BC1 port map ( PQ0 => PQ3_2, PF0 => PF3_2, PQ3 => PF0, PF3 => PF3, CLK => CLK, OE =>
OE, C0 => C4_2, I => I, AB => AB , AA => AA, D => D ( 15 downto 12 ), OVR => open, F3 => open, Z =>
open, C4 => C4_3, P => open, G => open, Y => DY ( 15 downto 12 ) );

end BOD ;
```

Аналогічним чином розробляються моделі блоків регістра мікрокоманд, тригера блокіровки, лічильника адрес мікрокоманд керуючого інвертора та буферного підсилювача з трьома станами. На заключному етапі розробки моделі схеми пристрою всі створені блоки збираються і з'єднуються внутрішніми сигналами. Вибираються та задаються також затримки для окремих блоків і вхідні параметри (кількість осередків МПП), а також визначається інтерфейс остаточної схеми.

Вхідними сигналами схеми рис. 1.2 є тактовий сигнал ***CLK*** (у явному вигляді генератор не задається) та сигнал ***"Пуск"***, а вихідними - значення сигналів шини ***Y (15 – 0)*** на виводі БОД.

```
entity  l1_BC1 is
 port (
        Y : out std_logic_VECTOR ( 15 downto 0 ) – вихідна шина даних
        CLK_IN : in bit ; -- тактовий сигнал
        START : in bit ; – сигнал "Пуск"
        ) ;
end l1_bc1 ;

architecture l1_bc1 of l1_bc1 is
compontnt MPP is
 generic ( ROM_size : natural : = 3 ) ;
```

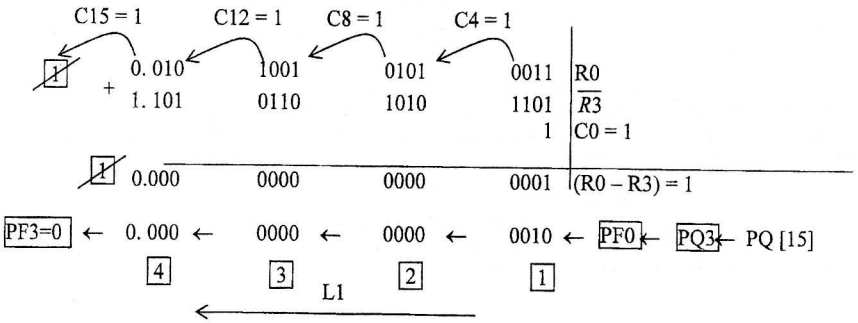
```
port (
        ADDR : in natural; -- адреса комірки
        MPP_CELL : out ROM_TYPE
        );
end component;
```

Аналогічним чином на VHDL - мові описуються компоненти та сигнали інших блоків системи.

На рис. 1.4 наведено фрагмент часової діаграми одного такту симуляції МПС при виконанні останньої (шостий) мікрокоманди ( Y : = 2 ( Y – C )). Для виконання тестового приклада було прийнято: A = +1025h; B = +2237h; C = + 2952h.

З аналізу часової діаграми випливає, що при надходжені чергового позитивного фронту тактового сигналу CLK (момент часу t1) у РМК записується обрана з МПП за адресою 6h чергова мікрокоманда. Через час $t_{ЗАТ}^{РМК} = 30\ нс$ (момент часу ta) на виході РМК формуються нові значення сигналів, які подаються на входи БОД: I_BC1 = 189h; AA = 3h; AB = 0h; C0 = 1; OE# = 0h.

БОД виконує операцію 2 ( R0 – R3 ) → R0 над операндами R0 = 10579 і R3 = 10578, які представлені у додатковому коді:




З приклада випливає, що перенос від молодшої (першої) секції до старшої (четвертої) секції поширюється послідовно. Затримка поширення переносу в одній секції складає 20 нс. З аналізу рис. 1.4 випливає, що правильний результат (рівний одиниці) формується на виходах молодшої секції Y (3 - 0) у момент часу t = 1270 нс, а на виходах самої старшої секції Y (15 – 12) - у момент часу t = 1330 нс.

Перед записом результату з шини F АЛП в регістр R0 РЗП на зсувачах ЗСВF і ЗСВQ здійснюється зсув подвійної довжини вмісту шини F та регістра PQ на один розряд ліворуч. При цьому у вивільнюваний праворуч розряд записується нуль з PQ [15]. Правильне значення спадаючого при зсуві розряду БОД виявляється на лінії PF3 самої старшої МПС у момент часу th.

Тривалість такту ($T = t_{12} + t_{23}$ ) сигналу CLK розрахована на основі співвідношення Д1.2 (див. додаток) при наступних значеннях затримок сигналів:
$t_{ЗАТ}^{РМК} = 30\ нс$ ; $t_{ЧТ}^{РЗП} = 10\ нс$; $t_{ЗАП}^{РА} = 10\ нс$; $t_{ЗАТ}^{МХР} = 5\ нс$; $t_{ЗАТ}^{АЛП} = 20\ нс$; $t_{ЗАП}^{РЗП} = 30\ нс$;
$t_{ЗАТ}^{ЗСВF} = 5\ нс$; $t_{ЗАТ}^{PQ} = 30\ нс$; $t_{ПІД}^{PQ} = 30\ нс$;

При цих значеннях було отримано, що $t_{12} = 140\ нс$ , $t_{23} = 30\ нс$ , $T = 170\ нс$.

Уся мікропрограма буде виконана за час $T_{МП} = 7 \cdot T = 1190\ нс = 1{,}19\ мкс$.

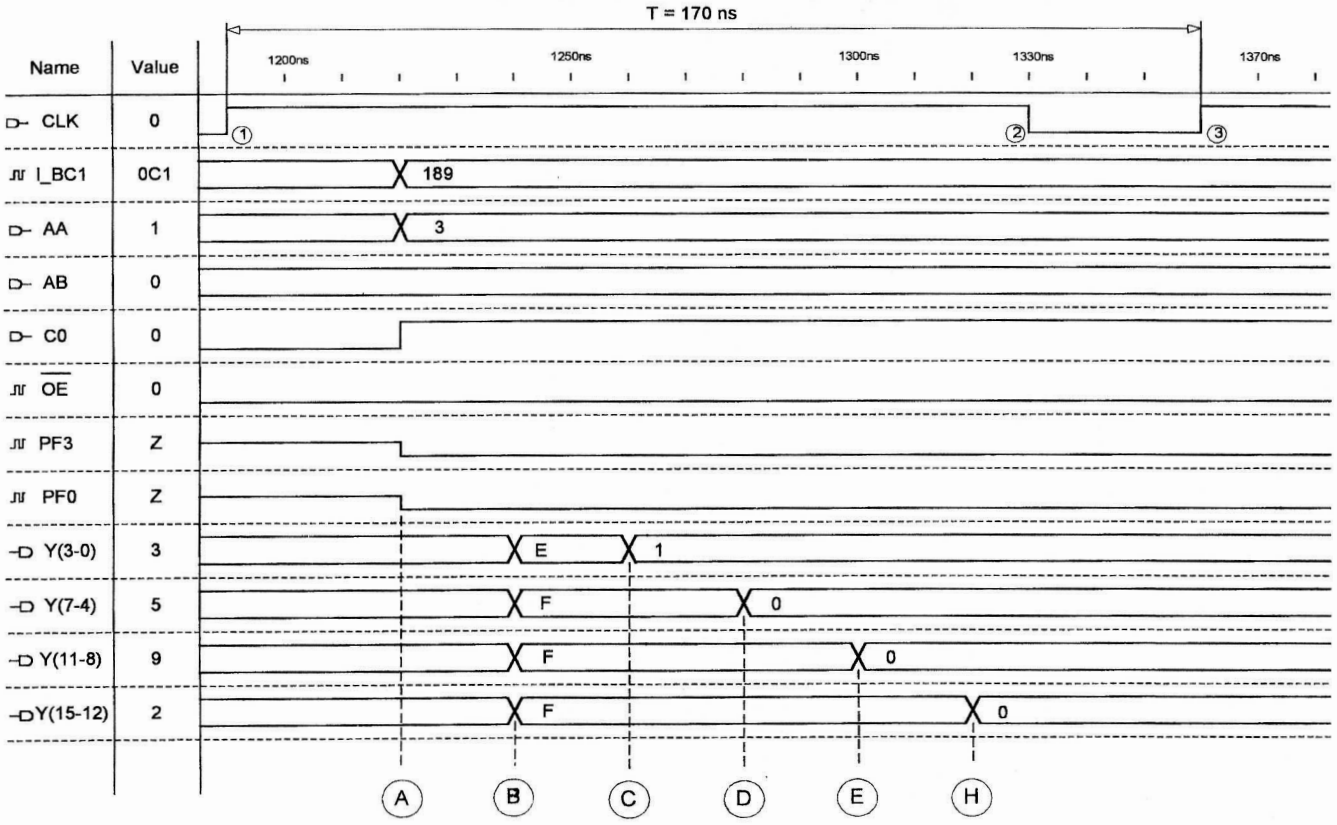



Рисунок 1.4 Часова діаграма одного такту роботи БОД

### 1.4. Зміст звіту

1. ГСМ та таблиця кодування реалізації заданої функції.
2*. Листінг VHDL – проекту.
3*. Результати експериментальних досліджень.
4. Розрахунок часу виконання мікропрограми.

**Примітка.** Пункти 2* та 3* виконуються факультативно.

### 1.5. Контрольні запитання

1. Нехай число A = - 5 розташоване у регістру R7 мікросхеми ВС1 і сформоване в додатковому коді. Як виконати зсув R7 в ВС1:
   А) - ліворуч на один розряд ? ;
   Б) - праворуч на один розряд ?

   Вказати, що при цьому необхідно підключити на клеми PF3, PF0, PQ3, PQ0 мікросхеми ВС1.
2. Яким чином у процесорі на базі двох секцій ВС1 (вісім розрядів) можливо виконати складання двох 16 – розрядних чисел ? Наведіть приклад виконання цієї операції.
3. У якому стані знаходяться клеми PF3 та PF0 зсувача шини F (3CBF) мікросхеми ВС1, коли зсув вітсутній ?
4. Яким чином організувати зсув вліво подвійної довжини на один розряд з використанням регістра R5 РЗП та регістра – акумулятора PQ ?
5. Яким чином на базі мікросхем ВС1 виконати складання чисел, які представлены в оберненому коді ? Наведіть алгоритм.
6. Яким чином можливо виявити переповнення розрядної сітки процесора ВС1, якщо виконуються арифметичні операції над числами:
   А) - у додатковому коді;
   Б) - у вигляді модулів.
7. Чим відрізняється поведінковий опис архітектури від структурного опису ?
8. Яким чином при симуляції вказують параметри вхідних сигналів ?
9. Яким чином організувати зсув праворуч подвійної довжини на один розряд із використанням регістра R3 РЗП та регістра – акумулятора PQ ?

**Лабораторна робота №2**

# ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ БАГАТОРОЗРЯДНОГО БОД З ВИКОРИСТАННЯМ СХЕМИ ПРИСКОРЕННОГО ПЕРЕНОСУ (СПП)

**МЕТА РОБОТИ:** придбати навички проектування та дослідження блоків обробки даних на основі процесорних секцій ВС1 та мікросхем прискоренного переносу ВР1.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

На основі мікропроцесорних секцій ВС1 (або ВС2) можуть бути виконані БОД з кількістю розрядів, яка кратна чотирьом (розрядність однієї секції). Недоліком такого багаторозрядного БОД з послідовним включенням секцій (наприклад, на 64 розряди) є велика затримка при формувані суми, так як перенос від однієї секції до іншої поширюється послідовно. В з'язку з чим, період такту синхросигналу для гарантування коректної роботи суматора доводиться збільшувати, що приводить до втрати продуктивності обчислювального приладу в цілому.

Цей недолік багаторозрядних БОД може бути значно зменшиний, якщо спільно з секціями ВС1 (або ВС2) використовувати схеми прискоренного переносу на основі мікросхем К1804ВР1 [1 – 5].

### 2.1. Послідовність виконання лабораторної роботи

1. Розробити функціональну схему багаторозрядного БОД на основі секцій ВС1 та ВР1.
2. Привести співвідношення та розрахувати час поширення переносу в БОД для схем із використанням СПП ВР1 та без його використання.

3*. Виконати налагодження VHDL – проекту у середовищі Active – HDL (цей пункт роботи виконується факультативно).

4. Скласти звіт про виконану лабораторну роботу та захистити його.
5. Дати відповіді на контрольні запитання.

### 2.2. Варіанти індивідуальних завдань

По – перше, для виконання цієї лабораторної роботи з табл. 2.1 необхідно вибрати кількість розрядів БОД, який треба спроектувати. Потім виконати всі пункти параграфа 2.1.

Варіанти розрядності БОД. Таблиця 2.1

| (N)mod10 | Кількість розрядів БОД |
|---|---|
| 0 | 8 |
| 1 | 12 |
| 2 | 16 |
| 3 | 20 |
| 4 | 24 |
| 5 | 28 |
| 6 | 32 |
| 7 | 40 |
| 8 | 44 |
| 9 | 48 |

## 2.3. Методичні вказівки до виконання лабораторної роботи

Важливою властивістю мікропроцесорного комплекту серії К1804 є можливість реалізації багаторозрядного БОД, який складається з секцій ВС1 (або ВС2) (рис. 2.1).

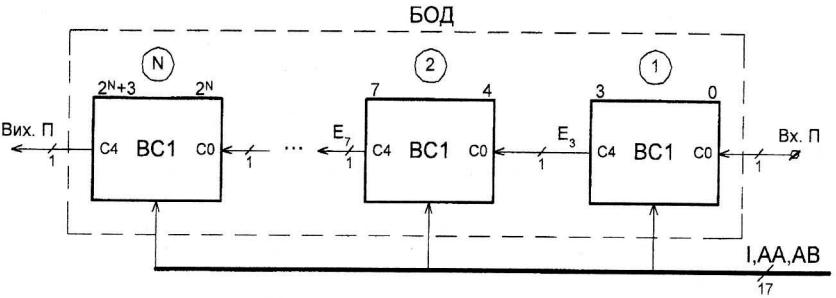



Рисунок 2.1. Організація багаторозрядного БОД з N секцій ВС1, які з'єднувані послідовно

Недоліком схеми рис. 2.1 є велика затримка сигналів суми в зв'язку з послідовним поширенням переносу від (ВхП) до (ВихП) БОД при виконанні арифметичних операцій. Таким чином, в такій схемі чим більше об'єднано мікропроцесорних секцій (МПС), тим повільніше робота БОД.

При розрахунку затримки переносу в багаторозрядної схемі необхідно урахувати слідуюче. В самій молодшій МПС перенос Е3 затримується відносно коду інструкції I та адресних кодів АА і АВ на 80 нс (ВС1). В інших секціях (середніх і старших) вихідной перенос затримується відносно вхідного значно менше - на 40 нс. В зв'язку з цим, в БОД із N секцій затримка суми і вихідного переносу відносно керуючих сигналів (I, АА, АВ) буде дорівнювати:

$$T_N = 80 + (N - 1) \cdot 40, \text{ (нс)}. \qquad (2.1)$$

Для зменшення часу поширення переносу в багаторозрядних БОД на основі секцій ВС1 (або ВС2) використовується схема прискореного переносу (СПП) на основі мікросхеми К1804ВР1 (див. додаток Д1.2 та Д1.4). Як керуючи сигнали в СПП використовуються сигнали $\overline{P_i}$ і $\overline{Gi}$ АЛП кожної МПС (сигнали підготовчих функцій). Якщо підготовча функція $\overline{Gi}$ приймає значення "істино", то на виході і –ої секції обов'язково виробляється сигнал переносу. Одиничне значення підготовчою функції поширення переносу $\overline{P_i}$ дозволяє передачу вхідного переносу і – ої секції в (і + 1) – у секцію. СПП виконана таким чином, що щодо часу появи функцій $\overline{P_i}$ і $\overline{Gi}$ формує одночасно вихідні переноси на своїх лініях CX, CY і CZ із затримкою 10 нс.

Схема підключення мікросхеми ВР1 при побудові БОД на 16 розрядів наведена на рис. 2.2. Вірні сигнали підготовчих функцій $\overline{P_i}$ і $\overline{Gi}$ на виходу секцій з'являться одночасно з затримкою відносно сигналів I, АА та АВ:

$$T_{P-G} = t_{ЧТ}^{РЗП} + t_{ЗП}^{РА} + t_{ЗАТ}^{MXR} + t_{ЗАТ}^{P-G} = 60 \text{ нс}. \qquad (2.2)$$

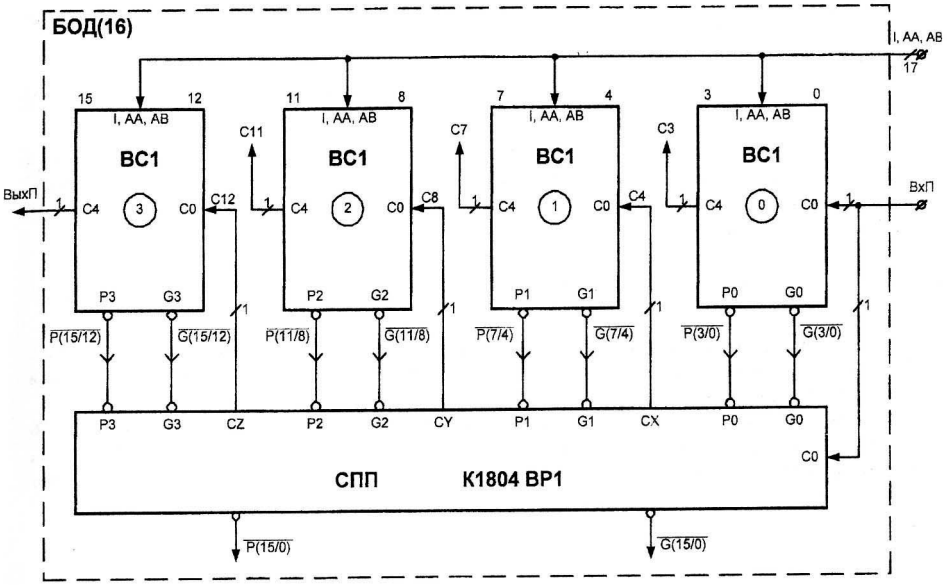



Рисунок 2.2. Організація БОД із 16 розрядів з використанням СПП

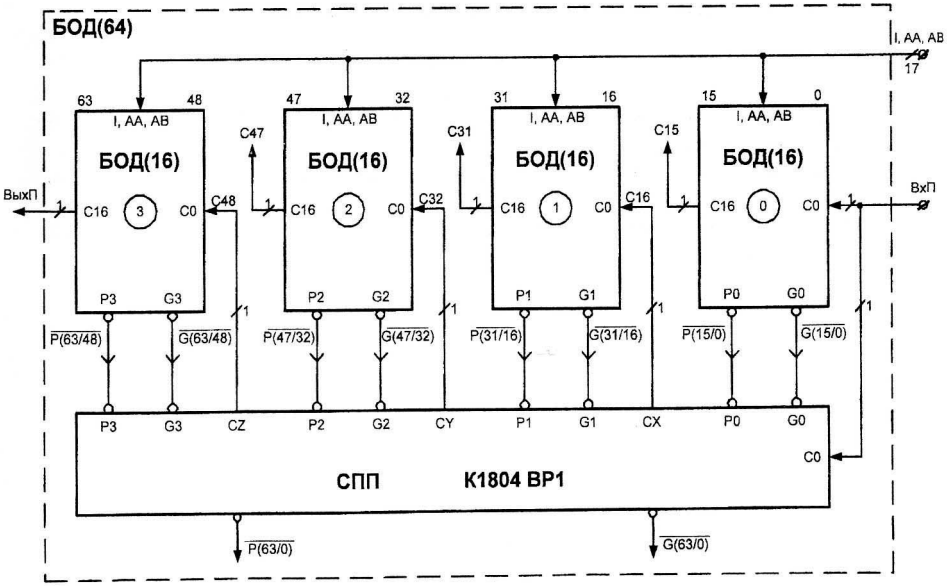



Рисунок 2.3. Організація БОД із 64 розрядів з використанням СПП

На лініях CX, CY і CZ СПП переноси формуються також одночасно після спрацьовування СПП:

$$T_{CX} = T_{CY} = T_{CZ} = t_{ЗАТ}^{СПП} = 10 \text{ нс.} \quad (2.3)$$

Сигнали CX, CY та CZ далі подаються на входи переносів другої (C4), третьої (C8) і четвертої (C12) секцій і після спрацьовування АЛП на вихідних лініях Y усіх МПС буде сформований коректний результат, а на лінії вихідного переносу - вірне значення переносу. При цьому

$$T_{ВИХ.П} = T_{P\text{-}G} + t_{ЗАТ}^{СПП} + t_{ЗАТ}^{АЛП} = (60 + 10) + 40 = 110 \text{ нс.} \quad (2.4)$$

З використанням п'яти СПП можливо реалізувати 64 – х розрядний БОД (рис. 2.3). Затримка поширення переносів у цьому випадку розраховується аналогічно (пропонується розрахувати самостійно).

На рис. 2.4 наведені результати симуляції в середовищі Active – HDL БОД на 16 розрядів з використанням СПП при підсумовуванні кодів A = 1111 1111 1111 1111 та B = 0000 0000 0000 0000 і переносі ВхП = 1. Часова діаграма в цілому коректно відбиває перехідні процеси в БОД, які наведені вище.

### 2.4. Зміст звіту

1. Функціональна схема БОД на основі мікросхем ВС1 та ВР1.
2. Формули для розрахунку затримки параметрів.
3*. Результати симуляції VHDL - проекту (виконується факультативно).

### 2.5. Контрольні запитання

1. За якими формулами формуються сигнали $\overline{P_i}$ та $\overline{G_i}$ секції ВС1 ? Яким чином сінтезувати ці формули ?
2. Як організувати БОД на 32 розряди ? Наведіть схему цього БОД на основі мікросхем ВС1 та ВР1. Які сигнали необхідно формувати на вхідних клемах СПП, які знаходяться в "повітрі" ? Як буде відбуватися поширення переносу у такому БОД ? Розрахуйте час затримки ВихП у такому БОД.

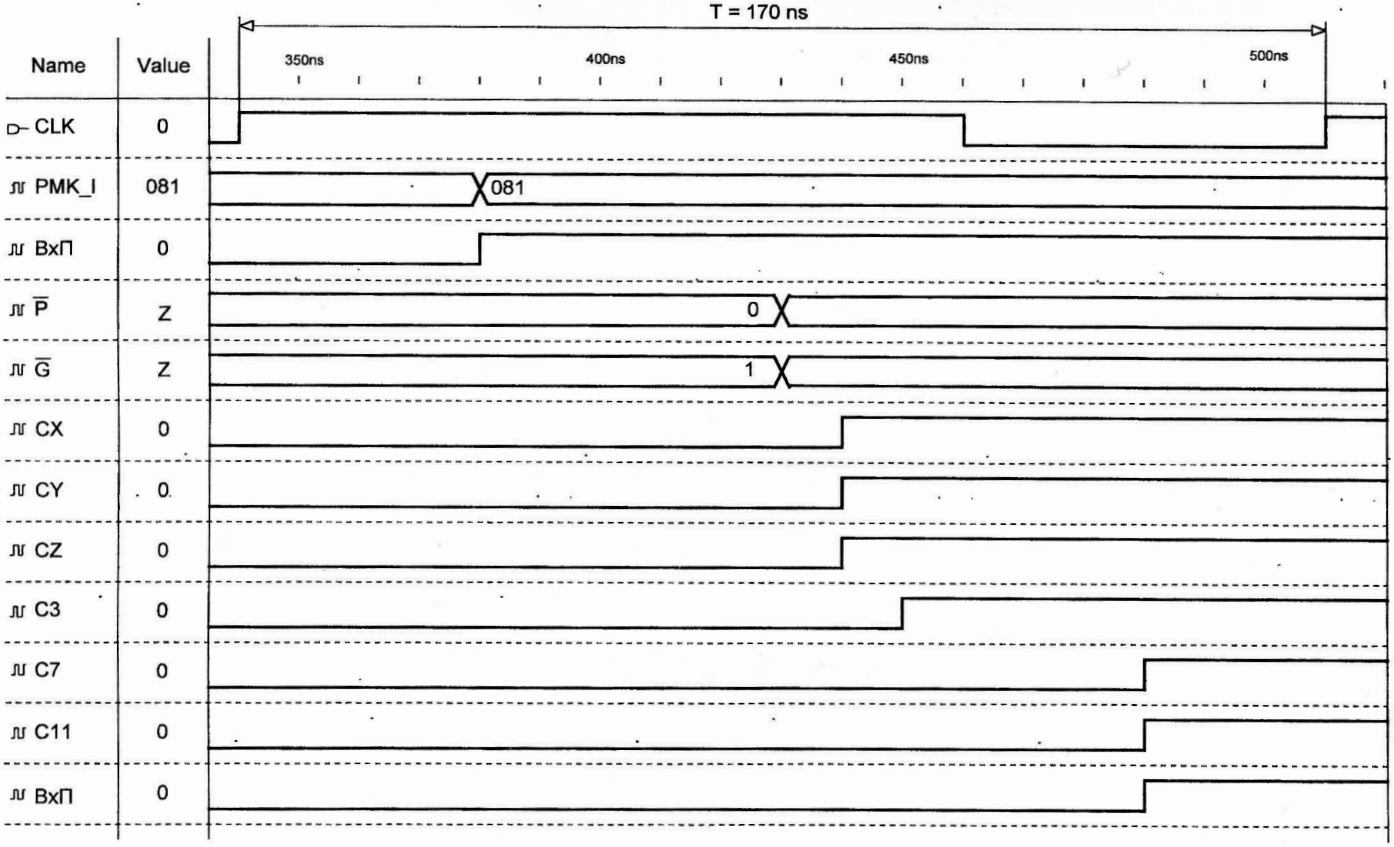



Рисунок 2.4 Часова діаграма симуляції 16 – розрядного БОД зі СПП

## Лабораторна робота №3

# РОЗРОБКА ТА СИМУЛЯЦІЯ БЛОКУ МІКРОПРОГРАМНОГО КЕРУВАННЯ (БМК) НА ОСНОВІ МІКРОСХЕМИ К1804ВУ4

**МЕТА РОБОТИ:** опанувати принципи функціонування мікропроцесорної секції керування адресою мікрокоманди (СКАМ) К1804ВУ4; придбати навички розробки і кодування мікроалгоритмів найпростіших функцій; навчитися розробляти імітаційні моделі БМК мовою VHDL, проводити їхню симуляцію в середовищі Active – HDL та аналізувати отримані результати.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

Проектування мікрообчислювачів на основі мікропроцесорних комплектів інтегральних схем серії К1804 з мікропрограмним керуванням жадає від розроблювача системного підходу - одночасного створення як апаратної частини, так і програмного забезпечення. Принцип мікропрограмного керування дозволяє простою заміною керуючої програми набудовувати мікрообчислювач на рішення нової задачі.

Мікросхема К1804ВУ4 призначена для реалізації блоків мікропрограмного керування. Основна функція ВУ4 складається у формуванні 12 – розрядних адрес мікрокоманд, яки знаходяться у мікропрограмної пам'яті (МПП).

### 3.1 Послідовність виконання лабораторної роботи

1. Вивчити з використанням літературних джерел [1 – 5] і методичних вказівок склад, призначення та алгоритми виконання команд мікросхемою ВУ4.
2. Для заданої ГСМ розробити таблицю кодування. У ГСМ забезпечити введення в регістрову пам'ять ВС1 операндів з використанням зовнішньої шини D.

3*. Розробити імітаційну модель мікросхеми ВУ4 та поместити її у бібліотеку К1804.lib.

4*. Розробити VHDL – модель мікрообчислювального пристрою (МОП) на основі ВУ4, двох секцій ВС1, МПП, РМК та регістра станів на DC – тригерах з некерованою синхронізацією.

5*. Виконати налагодження VHDL – проекту. Провести симуляцію роботи МОП в середовищі Active – HDL.

6. Скласти звіт про виконану лабораторну роботу та захистити його.
7. Дати відповіді на контрольні запитання.

**Примітка.** Пункти 3*, 4* та 5* виконуються факультативно.

### 3.2 Варіанти індивідуальних завдань

Скласти ГСМ роботи МОП з фрагментів мікроалгоритмів рис. 3.1, з'єднуючи їх у послідовності, зазначеної в табл. 3.1. Вхід у першу вершину ГСМ надходить з вершини "Початок", а вихід останньої - у вершину "Кінець".

### 3.3 Приклад виконання лабораторної роботи

При виконанні операцій "складання" та "віднімання" у ЦОМ в форматі з рухомою комою необхідно виконувати порівняння порядків операндів РА і РВ [1]. Будемо полагати, що порядки задані у додатковому коді, мають вісім розрядів і з них старший розряд - знаковий.

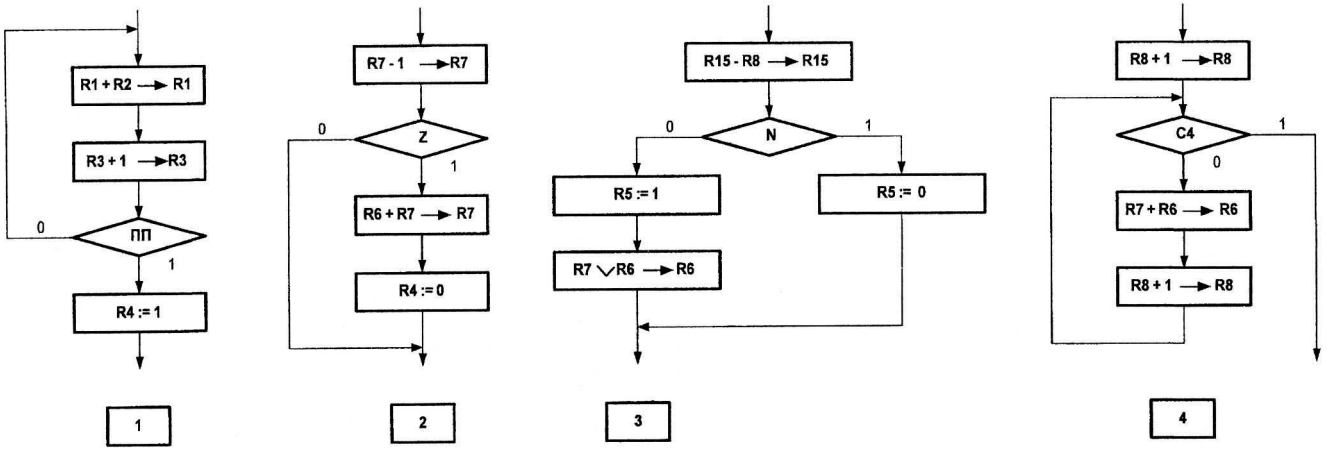



Рисунок 3.1. Фрагменти мікроалгоритмів для одержування початкової ГСМ

Порівняння порядків виконують шляхом віднімання від порядку РА порядку РВ та наступного аналізу ознак отриманого результату (рис. 3.2).

Варіанти завдань ГСМ. Таблиця 3.1

| (N)m16 | Послідовність фрагментів ГСМ |
|---|---|
| 0 | 1 → 2 → 3 → 4 |
| 1 | 4 → 1 → 2 → 3 |
| 2 | 3 → 4 → 1 → 2 |
| 3 | 2 → 3 → 4 → 1 |
| 4 | 1 → 2 → 4 → 3 |
| 5 | 3 → 1 → 2 → 4 |
| 6 | 4 → 3 → 1 → 2 |
| 7 | 2 → 4 → 3 → 1 |
| 8 | 1 → 3 → 2 → 4 |
| 9 | 4 → 1 → 3 → 2 |
| 10 | 2 → 4 → 1 → 3 |
| 11 | 3 → 2 → 4 → 1 |
| 12 | 2 → 1 → 3 → 4 |
| 13 | 4 → 2 → 1 → 3 |
| 14 | 3 → 4 → 2 → 1 |
| 15 | 1 → 3 → 4 → 2 |

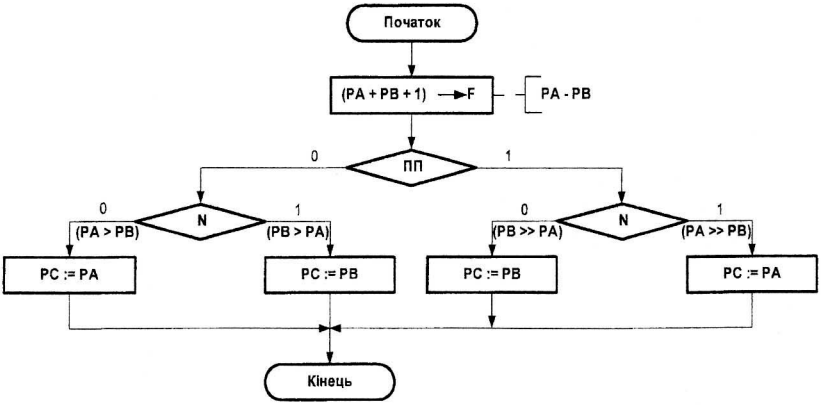



Рисунок 3.2. ГСА зрівнювання порядків

У випадку відсутності сигналу переповнення (ПП = 0) порядку результату РС привласнюють порядок більшого числа (РА або РВ) та приступають до етапу

вирівнювання мантис. При виникненні переповнення (ПП = 1) порядку результату привласнюють значення порядку більшого числа і завершують виконання операції.

Виконаємо розподіл регістрової пам'яті БОД: PA ~ R0; PB ~ R1; PC ~ R2. Початкова та перетворена ГСМ для розглянутого приклада представлені на рис. 3.3. У БОД ознаки переповнення ПП і знака результату операції N виробляються одночасно, але на вхід умови $\overline{CC}$, яка перевіряється мікросхемою ВУ4, вони повинні надходити в порядку черговості. Щоб не відбувалася втрата правильного значення, ознаки повинні бути записані в тригера регістра ознак (РП). У цьому випадку в одному такті можливо виконувати перевірку ознаки ПП (вихід тригера ТП), а в іншому - ознаки N (вихід тригера TN).

Функціональна схема МОП наведена на рис. 3.4. БОД складається з двох секцій ВС1, яки включені послідовно. У МПП записані 11 мікрокоманд. Перша мікрокоманда (МК0) мікропрограми знаходиться по нульовій адресі. Для переходу на цю мікрокоманду з пульта керування надходить сигнал "Start", по якому код нульової адреси через мультиплексор МХА надходить на вхід МПП. Обрана мікрокоманда записується по сигналу CLK у регістр мікрокоманд (РМК). З його виходу в блоки МОП надходять відповідні сигнали: у БОД ( I , AA , AB , C0 , $\overline{OE}$ ); у ВУ4 (MI , DA , $\overline{RLD}$ , CCE , $\overline{OEY}$ , C0); у МХ СС ( A2 , A1); у РП (UN , UII). Сигнал "Stop" виробляється в останній мікрокоманді виконуваної мікропрограми. По ньому відбувається блокування подачі тактового сигналу синхронізації CLK та МОП зупиняє свою роботу. При аналізі функціональної схеми будемо полагати, що регістри РП та РМК виконані на DC – тригерах зі спрацьовуванням по передньому фронту тактового сигналу.

Розряди A2 та A1 необхідні для керування роботою мультиплексора, який обирає для перевірки ВУ4 ознаки відповідно до табл. 3.2.

Робота мультиплексора МХ СС. Таблиця 3.2

| A2 | A1 | Вихід мультиплексора СС |
|---|---|---|
| 0 | 0 | Константа одиниці |
| 0 | 1 | Константа нуля |
| 1 | 0 | Вихід тригера ТП |
| 1 | 1 | Вихід тригера TN |

На основі перетвореної ГСМ (рис. 3.3,б) складена таблиці кодування роботи БОД (табл. 3.3) і ВУ4, РП та МХ СС (табл. 3.4). При складанні табл. 3.4 використана система мікрокоманд (див. табл. Д2.1 і рис. Д2.3 додатка). Мікропрограма починається з нульової адреси.

На рис. 3.5 наведений фрагмент часової діаграми симуляції для трьох мікрокоманд: МК2, МК3 та МК4 (див. рис. 3.3,б) для порядків PA = 3 та PB = 5. У момент часу $t_1$ в РМК записується код другої мікрокоманди (МК2). У БОД виконується задана операція (R0 – R1) → F та на його виходах формуються ознаки N і ПП через інтервал часу $t_{ЗАТ}^{РМК} + t_{ЗАТ}^{БОД}$. Тоді ж виробляються керуючі сигнали UN = 1 та UII = 1, під впливом яких спрацьовує схема прийому ознак до РП. До моменту часу $t_2$ (початку часу підготовки РП) перехідні процеси на D – входах тригерів РП повинні завершитися. В інтервалі часу ($t_2 \div t_3$) здійснюється підготовка до спрацьовування тригерів РП та РМК.

У момент часу $t_3$ спрацьовує регістр ознак РП та на його виходах формуються значення ознак (ТП = ПП та TN = N ), яки вироблені у мікрокоманді МК2. У РМК у цей же момент часу записується наступна мікрокоманда МК3, по якій у ВУ4 здійснюється перевірка ознаки ТП (операцією CJP), а в БОД - порожня операція (NOP). Для

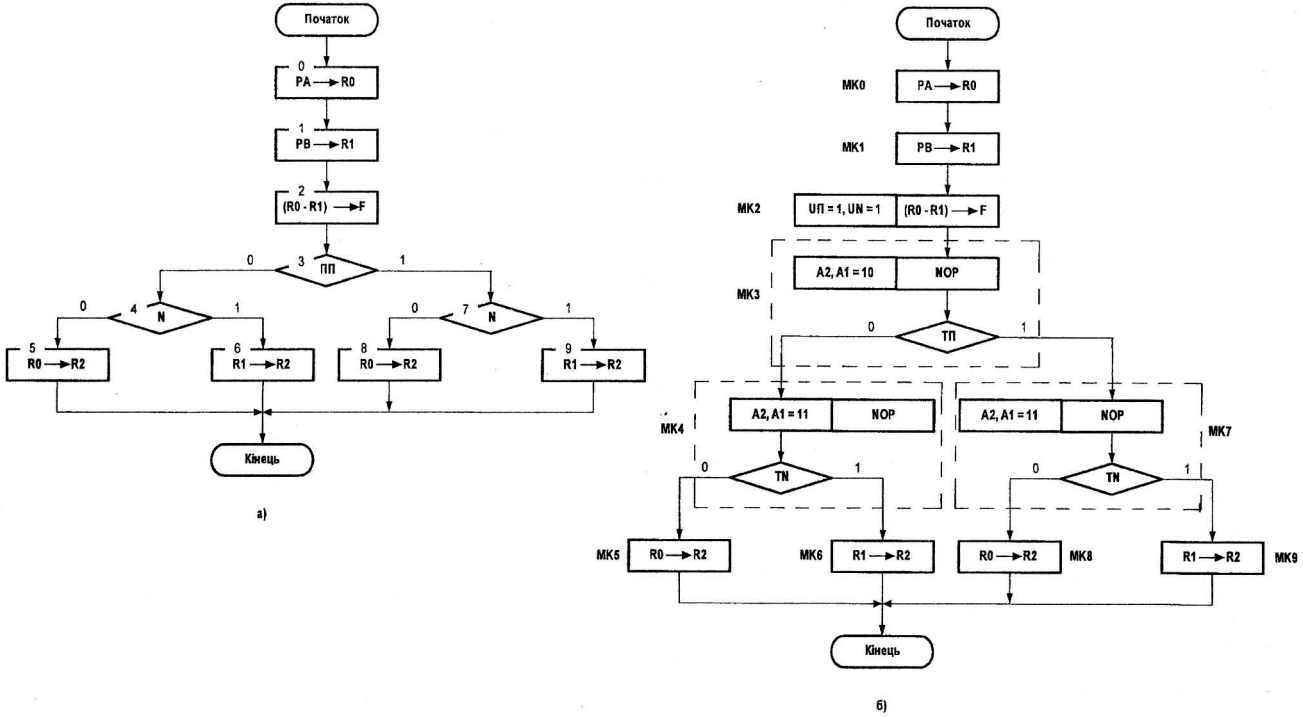



Рисунок 3.3. Початкова (а) та перетворювана (б) ГСМ

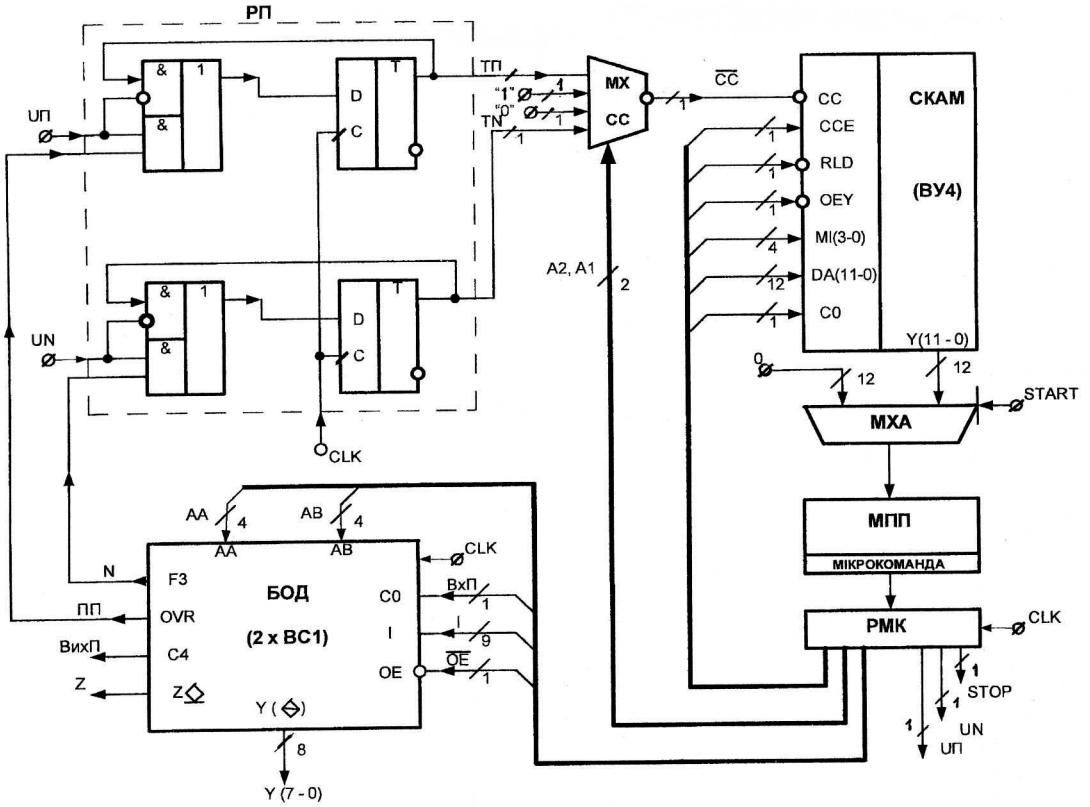



Рисунок 3.4. Функціональна схема МОП

Таблиця кодування БОД. Таблиця 3.3

| Мікрокоманда | Адреса МК | I8 - I6 | I5 – I3 | I2 – I0 | AA | AB | ВхП | $\overline{OE}$ | D | Операція ВС1 |
|---|---|---|---|---|---|---|---|---|---|---|
| МК0 | 0 | 3 | 0 | 7 | 0 | 0 | 0 | 0 | 003 | D ( PA ) → R0 |
| МК1 | 1 | 3 | 0 | 7 | 0 | 1 | 0 | 0 | 005 | D ( PB ) → R1 |
| МК2 | 2 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | * | ( R0 - R1 ) → F |
| МК3 | 3 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | * | NOP |
| МК4 | 4 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | * | NOP |
| МК5 | 5 | 3 | 0 | 4 | 0 | 2 | 0 | 0 | * | R0 → R2 |
| МК6 | 6 | 3 | 0 | 4 | 1 | 2 | 0 | 0 | * | R1 → R2 |
| МК7 | 7 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | * | NOP |
| МК8 | 8 | 3 | 0 | 4 | 0 | 2 | 0 | 0 | * | R0 → R2 |
| МК9 | 9 | 3 | 0 | 4 | 1 | 2 | 0 | 0 | * | R1 → R2 |
| МК10 | 10 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | * | NOP |



Таблиця кодування ВУ4, МХ СС та РП. Таблиця 3.4

| Мікрокоманда | Адреса МК | MI | $\overline{RLD}$ | CCE | $\overline{OEY}$ | DA | C0 | A2 | A1 | UII | UN | Stop | Операція ВУ4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| МК0 | 0 | E | 1 | * | 0 | * | 1 | * | * | 0 | 0 | 0 | CONT |
| МК1 | 1 | E | 1 | * | 0 | * | 1 | * | * | 0 | 0 | 0 | CONT |
| МК2 | 2 | E | 1 | * | 0 | * | 1 | * | * | 1 | 1 | 0 | CONT |
| МК3 | 3 | 3 | 1 | 1 | 0 | 7 | 1 | 1 | 0 | 0 | 0 | 0 | CJP |
| МК4 | 4 | 3 | 1 | 1 | 0 | 6 | 1 | 1 | 1 | 0 | 0 | 0 | CJP |
| МК5 | 5 | 3 | 1 | 0 | 0 | 10 | 1 | * | * | 0 | 0 | 0 | CJP |
| МК6 | 6 | 3 | 1 | 0 | 0 | 10 | 1 | * | * | 0 | 0 | 0 | CJP |
| МК7 | 7 | 3 | 1 | 1 | 0 | 9 | 1 | 1 | 1 | 0 | 0 | 0 | CJP |
| МК8 | 8 | 3 | 1 | 0 | 0 | 10 | 1 | * | * | 0 | 0 | 0 | CJP |
| МК9 | 9 | E | 1 | * | 0 | * | 1 | * | * | 0 | 0 | 0 | CONT |
| МК10 | 10 | * | 1 | * | 0 | * | 1 | * | * | 0 | 0 | 1 | NOP |

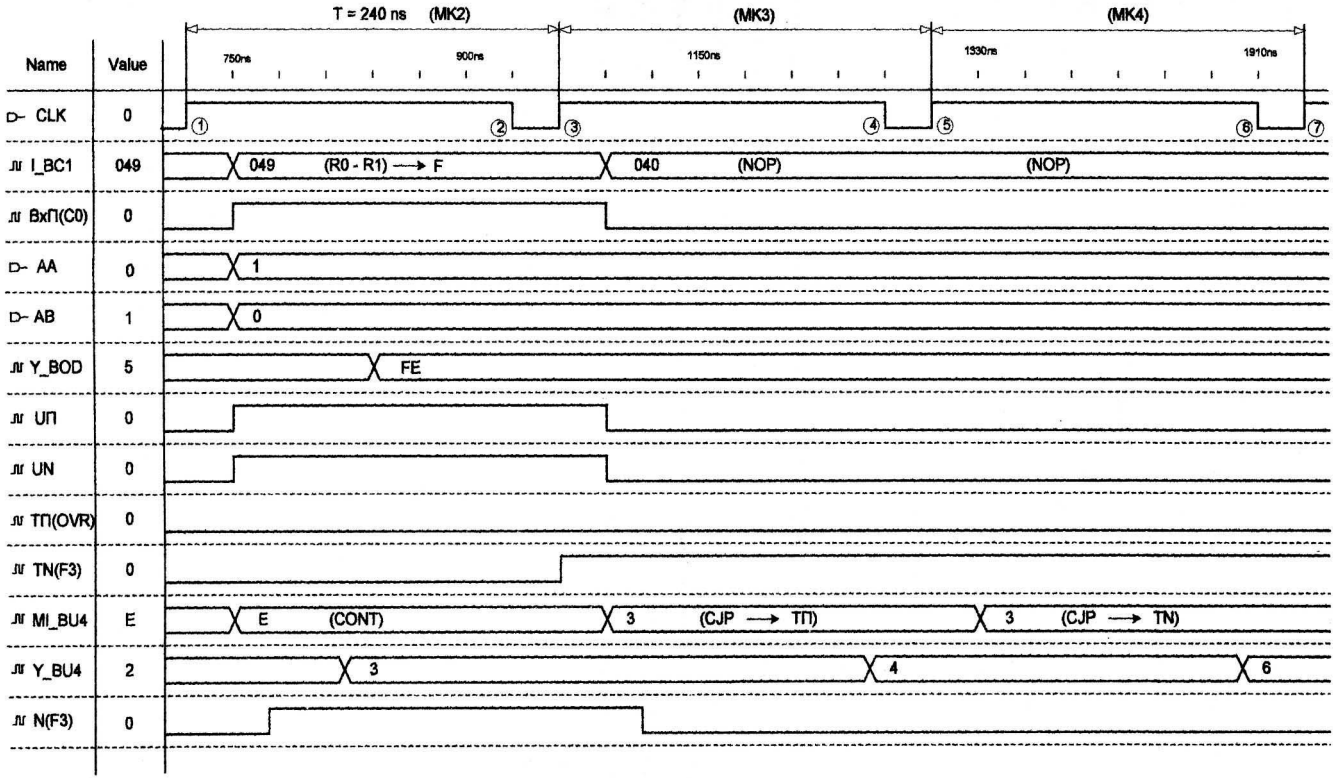



Рисунок 3.5 Часова діаграма симуляції 2, 3 та 4 мікрокоманд

формування на вході $\overline{CC}$ ВУ4 ознаки ТП на мультиплексор МХ СС варто подати A2 = 1 і A1 = 0 (див. табл. 3.2). Операція NOP означає, що БОД може виконувати у цьому такті будь-яку операцію, однако запису результату в РЗП не повинно відбуватися. У залежності від значення ТП ВУ4 сформує адресу мікрокоманди МК4 (якщо ТП = 0), або адресу мікрокоманди МК7 (якщо ТП = 1).

В інтервалі часу $(t_3 \div t_4)$ спрацьовують схеми РП, МХ СС, ВУ4 та МПП. До моменту часу $t_4$ перехідні процеси в них повинні завершитися для того, щоб під час $(t_4 \div t_5)$ здійснилася підготовка до спрацьовування РМК.

У момент часу $t_5$ спрацьовує РМК і в нього записується нова мікрокоманда (МК4 або МК7) перевірки ознаки TN. Ця мікрокоманда виконується аналогічно розглянутій раніше мікрокоманді МК3.

Тривалість одного такту роботи (параметр тактового сигналу CLK) може бути визначена з формули:

$$T = t_{12} + t_{23}\,, \tag{3.1}$$

де $t_{12} \geq \max\{t_{ЗАТ}^{РП} + t_{ЗАТ}^{МХСС} + t_{ЗАТ}^{ВУ4} + t_{ЗАТ}^{МХА} + t_{ЗАТ}^{МПП};\ t_{ЗАТ}^{РМК} + t_{ЗАТ}^{БОД} + t_{ЗАТ}^{СХ.ПР.}\}$;

$t_{23} \geq \max\{t_{під}^{РМК},\ t_{під}^{РП},\ t_{ЗАП}^{РЗП}\}$;

При виконанні симуляції МОП приймалися наступні значення параметрів затримок у роботі блоків: затримки часів спрацьовування РМК $t_{ЗАТ}^{РМК}$ = 30 нс, РП $t_{ЗАТ}^{РП}$ = 30 нс; час затримки в БОД з двох секцій $t_{ЗАТ}^{БОД}$ = 100 нс; час спрацьовування мультиплексора МХ СС $t_{ЗАТ}^{МХСС}$ = 20 нс; час затримки спрацьовування ВУ4 $t_{ЗАТ}^{ВУ4}$ = 70 нс; час затримки спрацьовування МПП $t_{ЗАТ}^{МПП}$ = 70 нс; час затримки запису інформації в РЗП ВС1 $t_{ЗАП}^{РЗП}$ = 30 нс; час затримки схеми прийому РП $t_{ЗАТ}^{СХ\ ПР.}$ = 20 нс; час підготовки тригерів РП та РМК $t_{ЗАТ}^{РМК} = t_{під}^{РП}$ = 30 нс.

При цих значеннях були отримані наступні значення параметрів сигналу CLK: $t_{12}$ = 210 нс; $t_{23}$ = 30 нс; $T$ = 240 нс.

### 3.4 Зміст звіту

1. ГСМ (початкова та перетворена) і таблиця кодування роботи МОП (для БОД, ВУ4, МХ СС та інших блоків).
2. Листінг VHDL – проекту (виконується факультативно).
3. Результати експериментальних досліджень (виконується факультативно).

### 3.5 Контрольні запитання

1. Яким чином розрахувати тривалість такту роботи МОП за умовою, що в його складі вітсутній регістр ознак РП ?
2. Поясніть алгоритм виконання інструкцій ВУ4: **JMAP, JRP** та **PUSH**.
3. Яким чином з використанням мікросхеми ВУ4 здійснити безумовний перехід по заданій адресі мікрокоманди ?

**Лабораторна робота №4**

# ПРОГРАМУВАННЯ ПІДПРОГРАМ ТА ЦИКЛІВ З ВИКОРИСТАННЯМ МІКРОПРОЦЕСОРНОЇ СЕКЦІЇ К1804ВУ4

**МЕТА РОБОТИ:** придбати практичні навички роботи зі стеком при наявності в мікропрограмі підпрограм та циклів; навчитися розробляти VHDL – моделі мікропрограм з використанням стекових операцій, виконувати їхню симуляцію в середовищі Active – HDL та проводити аналіз отриманих результатів.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

Механізм підпрограм дозволяє звертатися з різних місць основної головної програми до одного і того же фрагменту. При цьому зменшується кількість комірок пам'яті, яку займає програма.

У багатьох випадках одна підпрограма може викликати другу, яка, в свою чергу, може викликати третю підпрограму і так далі. Для коректної роботи пристрою необхідно організувати "вкладення" адрес повернення, які служать для поєднання окремих частин програми.

Організувати вибірку адресів повернення у черзі, яка зворотна їх надходженню (дисципліна обслуговування типу LIFO), забезпечує стекова пам'ять, яка є в складі мікросхеми К1804ВУ4 [1 – 5].

Для керування роботою стека ВУ4 використовуються наступні керуючі сигнали:

- сигнал завантаження / витяг **PUP** (Push / Pop), якій задає тип операції зі стеком; він також забезпечує збільшення або зменшення вмісту показника стеку SP на одиницю;
- сигнал дозволу доступу до стеку **FE#** (File Enable); видається завжди при звертанні до стеку з метою виконання операції завантаження або витягу.

Показник стеку **SP** завжди задає адресу мікрокоманди, яка записана в стек останньою. Стекова пам'ять є в складі мікропроцесорних секцій К1804ВУ4 та К1804ВУ1 [1, 2], що значно полегшує організацію і виконання підпрограм і циклів.

### 4.1 Послідовність виконання лабораторної роботи

1.Вивчити з використанням літературних джерел [1 – 5] та наступних методичних вказівок інструкції мікросхеми ВУ4, які використовуються при роботі зі стеком: **CJS, CRTN, PUSH, JSPR, RFST, LOOP, CJPP.**

2. Оформити фрагменти мікроалгоритмів з лабораторної роботи №3 (див. рис. 3.1) у вигляді основної програми та трьох підпрограм, з'єднуючи їх у послідовності, яка зазначена в табл. 3.1. Основна програма повинна здійснювати виклик першої підпрограми, та, у свою чергу, - другий, а друга - третьої підпрограми. Для одного або двох фрагментів підпрограм організувати циклові ділянки з використанням стекових операцій.

3. Розробити ГСМ та таблицю кодування (ТК).

4*. З використанням VHDL - моделі МОП з лабораторної роботи №3 виконати симуляцію розробленої ТК (цей пункт роботи виконується факультативно).

5. Скласти звіт про виконану лабораторну роботу та захистити його.

6. Дати відповіді на контрольні запитання.

### 4.2 Варіанти індивідуальних завдань

Варіанти індивідуальних завдань (фрагменти мікроалгоритмів) треба взяти з лабораторної роботи за №3.

### 4.3 Приклад виконання лабораторної роботи

На рис. 4.1 наведена схема взаємодії основної програми та трьох підпрограм, які послідовно викликають одна одну. У третій підпрограмі є циклічна ділянка програми, яка обробляється з використанням стекових операцій

В основній програмі перед зверненням до першої підпрограми відбувається початкова установа показника стеку (SP : = 0). Для цієї дії може бути використана інструкція мікросхеми ВУ4 **JZ** з кодом I = 0000 (перехід за нульовою адресою).

У другій мікрокоманді основної програми розташована інструкція **CJS** з кодом I = 0001, по якій відбувається перехід до першої підпрограми з початковою адресою 120. Одночасно у першу комірку стеку завантажується адреса 3 для повернення до основної програми. За адресою 125 в першій підпрограмі розташована інструкція **CRTN** з кодом I = 1010, за якою зі стека буде витягнута адреса 3 і таким чином буде здійснене повернення до основної програми.

Аналогічним чином виконується робота з другою та третьою підпрограмами. Схема взаємодії основної програми та підпрограм, а також зміна станів комірок стеку наведені на рис. 4.1.

Цикл у розглядаємому прикладі знаходиться у третій підпрограмі. Для цього по інструкції **PUSH** з кодом I = 0100 (розташована за адресою 750) виконується завантаження початкової адреси 751. Одночасно у лічильник РА / СТ ВУ4 завантажується число, яке на одиницю менше кількісті повторень циклу. Наприкінці циклу за адресою 754 розташована інструкція **RFCT** з кодом I = 1000. При її виконанні відбувається порівняння вмісту лічильника РА / СТ з нулем. Якщо РА / СТ ≠ 0, відбувається мікрооперація РА / СТ : = РА / СТ - 1, після чого з верхньої частини стеку витягається адреса 751 (таким чином виконується продовження циклу). У разі коли РА / СТ = 0, відбувається вихід з циклу. У цьому випадку керування приймає на себе мікрокоманда з адресою 755. Крім того, відбувається зменшення показника стеку SP на одиницю.

Таблицю кодування розглянутого прикладу пропонується скласти самостійно.

### 4.4 Зміст звіту

1. ГСМ та таблиця кодування.
2*. Листінг VHDL – проекту.
3*. Результати експериментальних досліджень.
4. Діаграма зміни вмісту стеку у процесі виконання мікропрограми.

**Примітка.** Пункти 2 та 3 звіту виконуються факультативно.

### 4.5 Контрольні запитання

1. З якою метою треба виконувати початкове установлення показника стеку ВУ4 ? Чому в стеку мікросхеми ВУ1 ця операція вітсутня ?

2. Що відбудеться, коли усі пьять комірок стеку ВУ4 будуть заповнені ?

3. Перелічить послідовність дій, яку необхідно виконати для організації цикла з використанням стеку ?

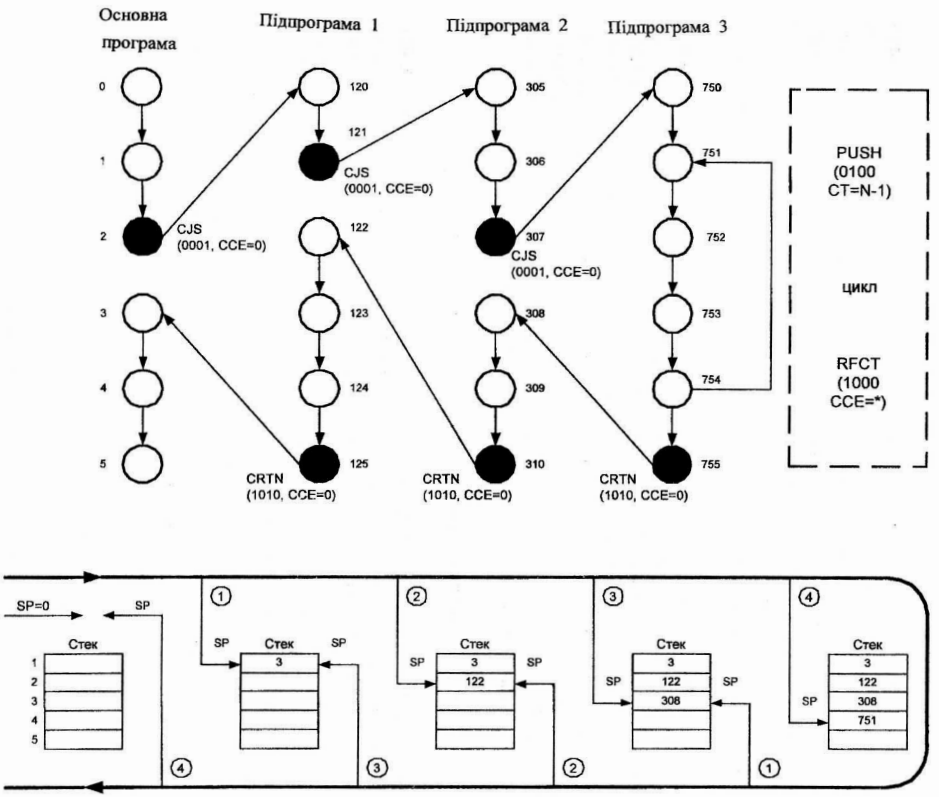



Рисунок 4.1. Приклад побудови підпрограм

## Лабораторна робота №5

# ПРОЕКТУВАННЯ ТА СИМУЛЯЦІЯ КОНВЕЄРНИХ МІКРООБЧИСЛЮВАЛЬНИХ ПРИСТРОЇВ З ОДНОФАЗНОЮ ТА БАГАТОФАЗНОЮ СИНХРОНІЗАЦІЄЮ

**МЕТА РОБОТИ:** придбати навички розробки, симуляції та дослідження пристроїв конвеєрної архітектури з використанням мікропроцесорів серії К1804; навчитися проводити аналіз та розраховувати часові параметри системи синхронізації.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

Одним з ефективних способів підвищення продуктивності мікрообчислювальних пристроїв з мікропрограмним керуванням (МОП) є використання конвеєрного методу обробки інформації [1, 2]. Конвеєрна обробка дозволяє в одному такті роботи МОП сполучити в часі роботу окремих блоків. Наприклад, БОД виконує поточну мікрокоманду, а з МПП вибирається вже наступна мікрокоманда. Така робота МОП забезпечується за рахунок включення додаткових регістрів у загальний тракт проходження сигналів, що дозволяє зберігати на виходах цих регістрів результати обробки різних мікрокоманд в одному такті.

Якщо, наприклад, у тракті передачі сигналів МОП є три регістри (мікрокоманд, адреси, ознак БОД), то в процесі конвеєрної обробки сполучаються операції, які відносяться до трьох мікрокоманд. Така архітектура МОП дозволяє істотно підвищити швидкодію у випадку виконання лінійних ділянок мікропрограм. При обробці мікрокоманди умовного переходу для коректної роботи потрібно затримати перевірку умови в БМК на два такти шляхом введення в схему алгоритму порожніх операторных вершин (NOP). Це є недоліком конвеєрної архітектури: при виконанні умовних переходів ускладнюється мікропрограмування і знижується швидкодія пристрою.

### 5.1. Послідовність виконання лабораторної роботи

1. Розробити функціональну схему МОП на основі двох секцій ВС1, ВУ4, МПП. Припустимо, що у склад МОП входять також три регістри, які виконані на DC - тригерах (працюють по передньому фронту позитивних тактових сигналів): мікрокоманд (РМК), адреси мікрокоманди (РА), ознаків результату БОД (РПП).
2. Для заданої початкової ГСА розробити ГСМ та таблицю кодування з коментарями у двох випадках:
   - синхронізація МОП виконується одним тактовим сигналом CLK;
   - в МОП використовується багатофазна синхронізація.
3. Розрахувати часові параметри систем синхронізації (однофазної та багатофазної) та виконати їх порівняльний аналіз.

4*. Виконати налагодження VHDL – проекту конвеєрної архітектури МОП та провести симуляцію його роботи (цей пункт роботи виконується факультативно).

5. Скласти звіт про виконану лабораторну роботу та захистити його.
6. Дати відповіді на контрольні запитання.

### 5.2. Варіанти індивідуальних завдань

Як початкову ГСА треба використовувати перші два мікроалгоритма з лабораторної роботи за № 3. Інші необхідні дані треба вибрати самостійно.

### 5.3. Методичні вказівки до виконання лабораторної роботи

Виконаємо аналіз функціонування мікрообчислювального пристрою (МОП) з використанням структури, наведеної на рис. 5.1, а [1, 2]. До її складу входять СКАМ на основі мікросхеми ВУ4, БОД на основі мікросхем ВС1 (або ВС2), МПП, мультиплексор вибору умови МХСС та три конвеєрних регістри: мікрокоманд (РМК), стану ознак переходу (РПП) і адреси мікрокоманди (РА). Припустимо, що усі регістри МОП виконані на DC – тригерах і спрацьовують під час дії переднього фронту (н → в) тактового імпульсу синхронізації CLK.

Звичайний цикл роботи МОП починається з надходження сигналу CLK, після чого відбувається передача сигналів від одних компонентів пристрою до інших. На рис. 5.1, б наведені часові діаграми виводів поширення сигналів щодо конвеєрних регістрів. З аналізу цих діаграм можуть бути складені співвідношення для розрахунку параметрів тактового сигналу синхронізації:

$$T \geq t_{12} + t_{23}, \qquad (5.1)$$

$$\text{де } t_{12} \geq \max\left[\left(t_{ЗАТ}^{БОД} + t_{ЗАТ}^{РМК}\right), \left(t_{ЗАТ}^{МПП} + t_{ЗАТ}^{РА}\right), \left(t_{ЗАТ}^{РПП} + t_{ЗАТ}^{МХСС} + t_{ЗАТ}^{СКАМ}\right)\right],$$

$$t_{23} \geq t_{під}^{ТР}$$

Якщо мікропрограма лінійна (не містить умовних вершин), то в одному такті БОД працює з і – ю мікрокомандою (МК), МПП - з (і + 1) – ю, а СКАМ - з (і + 2) – ю мікрокомандами. У цьому випадку має місце конвеєрна обробка послідовності мікрокоманд. При запуску МОП необхідно, щоб БОД в перших двох тактах сигналу CLK виконав команду NOP (почекав, поки надійде перша мікрокоманда мікропрограми).

Інакше здійснюється обробка мікрокоманд умовних переходів. Як приклад, розглянемо фрагмент ГСМ наведеної на рис. 5.2, а. У п'ятій мікрокоманді (МК5) у БОД виконується арифметична операція інкрементування вмісту регістра R2. При цьому перевірка результату операції (вихідного переносу С4) повинна бути затримана на два такти (рис. 5.2, б). Очевидно, перед перевіркою умови переходу (С4) необхідно в перетворену ГСМ вставити в поле функції БОД дві порожні операції (NOP). Це недолік даної архітектури, однак тривалість такту (5.1) буде мінімальної в порівнянні з іншими архітектурами МОП.

Іншим варіантом синхронізації МОП є застосування генератора багатофазних тактових імпульсів (ГБТІ). Як приклад, розглянемо синхронізацію пристрою (рис. 5.1,а) за допомогою трьох сигналів (рис. 5.3): CLK1, який подається на РМК і БОД, CLK2 - на РПП і СКАМ та CLK3 - на РА. У цьому випадку тривалість одного такту роботи системи може бути визначена із співвідношень:

$$T \geq t_{12} + t_{23} + t_{34} + t_{45} + t_{56} + t_{67}, \qquad (5.2)$$

$$\text{де} \quad t_{12} \geq t_{ЗАТ}^{РМК} + t_{ЗАТ}^{БОД}; \quad t_{34} \geq t_{ЗАТ}^{РПП} + t_{ЗАТ}^{МХСС} + t_{ЗАТ}^{СКАМ}; \quad t_{56} \geq t_{ЗАТ}^{РА} + t_{ЗАТ}^{МПП};$$

$$t_{23} \geq t_{під}^{РПП}; \quad t_{23} \geq t_{під}^{РА}; \quad t_{23} \geq t_{під}^{РМК}.$$

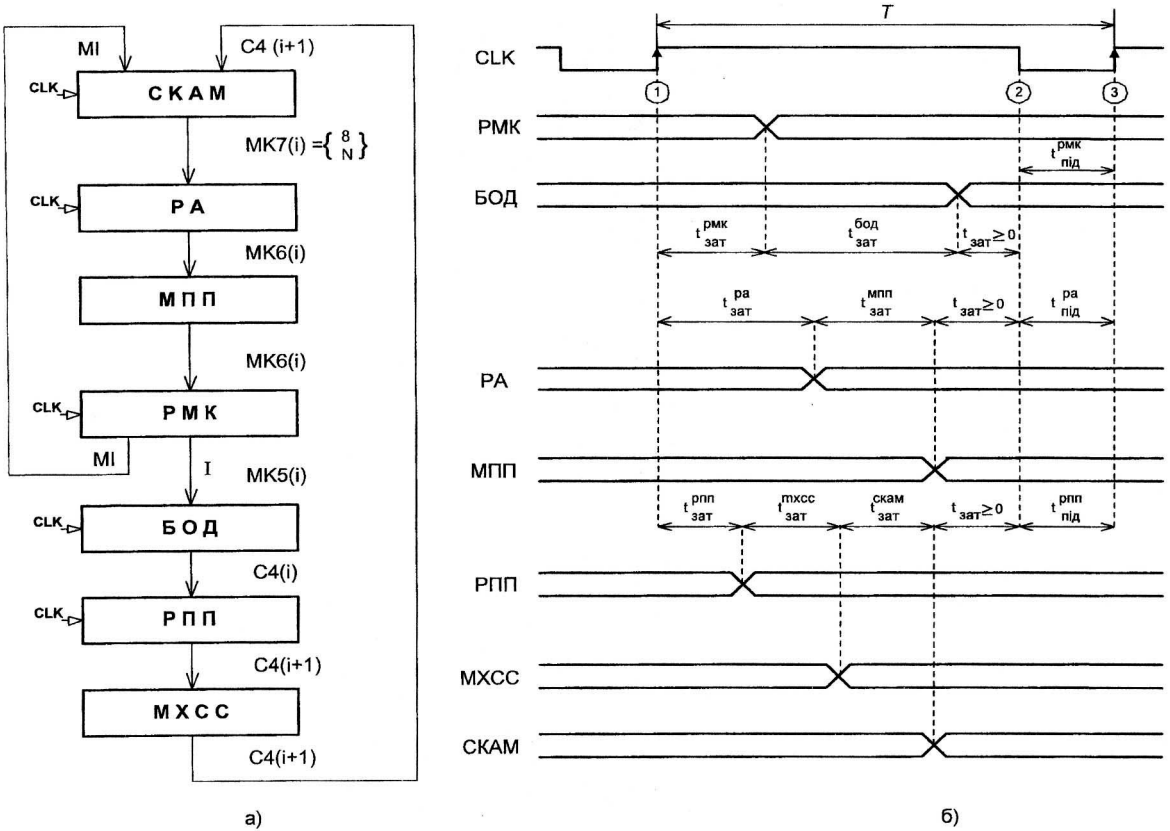



Рисунок 5.1. Конвеєрна структура (а) та часова діаграма (б) перехідних процесів

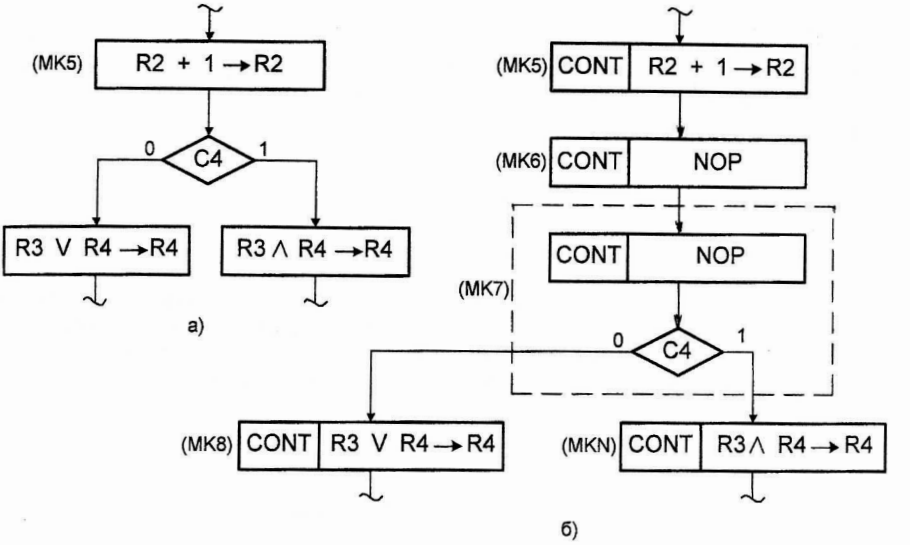



Рисунок 5.2. Фрагмент ісходної (а) та перетворюваної ГСМ (б)

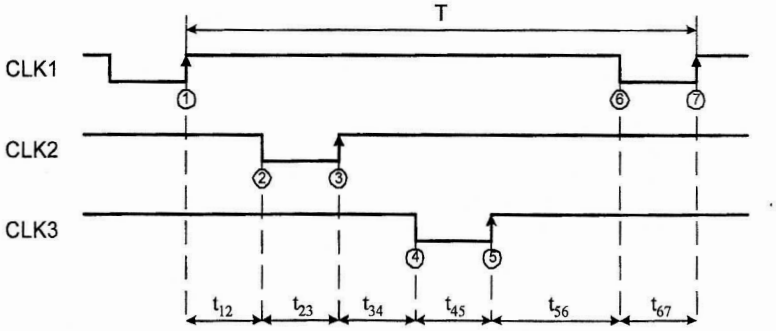



Рисунок 5.3. Часова діаграмма формування трьох синхронізуючих сигналів

При цьому в одному такті формується умова (наприклад, С4) на виході БОД і перевірка її СКАМ ВУ4. Таким чином, вводити порожні вершини і будувати перевірку ГСМ в цьому випадку немає необхідності. Мікропрограмування легшає в порівнянні з варіантом однофазної синхронізації. Недолік багатофазної системи синхронізації полягає в необхідності реалізації складної схеми ГБТІ.

### 5.4. Зміст звіту

1. Функціональна схема конвеєрного МОП (з визначенням призначення блоків та сигналів.
2. Початкова ГСА, перетворена ГСМ з порожніми вершинами та таблиці кодування роботи МОП з однофазної та багатофазною системами синхронізації
3. Часові діаграми роботи МОП при виконанні одного такту мікрокоманд з умовним та послідовним переходами, а також розрахунок часових інтервалів синхронізації.

4*. Результати симуляції VHDL – проекту (виконується факультативно).

### 5.5. Контрольні запитання

1. Як буде працювати МОП, коли в тракті передачі сигналів не буде жодного з регістрів ?
2. В архітектуре МОП є два регістри (РМК та РПП), які виконані на JK – тригерах (працюють по задньому фронту позитивного сигналу CLK). Приведіть формули для розрахунку часових параметрів синхросигналу з використанням однофазної системи синхронізації.
3. Запропонуйте схему побудови генератора трьохфазної системи синхронізації, якій може бути використовуваний в конвеєрному МОП.
4. Яким чином в VHDL – проекті організувати обробку паралельних процесів ? Наведіть відповідні приклади.
5. Яким чином буде реагувати активний процес у разі зміни стану функцій чутливості ?
6. Як виконати ініціалізацію роботи трьохфазного генератора тактових сигналів в середовищі Active – HDL ?
7. В архітектуре МОП є два регістри (РМК та РА), які виконані на DC – тригерах (працюють по передньому фронту позитивного сигналу CLK). Приведіть формули для розрахунку часових параметрів синхросигналів з використанням однофазної та двофазної системи синхронізації.
8. В архітектуре МОП є два регістри (РМК та РПП), які виконані на DE – тригерах типу "заскочка". Приведіть формули для розрахунку часових параметрів синхросигналів з використанням багатофазної системи синхронізації.
9. В архітектуре МОП є три регістри (РМК, РА та РПП), які виконані на DE – тригерах типу "заскочка". Приведіть формули для розрахунку часових параметрів синхросигналів з використанням багатофазної системи синхронізації.

**Лабораторна робота №6**

# ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ ПРИСТРОЮ МНОЖЕННЯ ЦІЛИХ ЧИСЕЛ З ВИКОРИСТАННЯМ МІКРОПРОЦЕСОРНИХ СЕКЦІЙ ВС1 І ВУ4

**МЕТА РОБОТИ:** придбати навички розробки та мікропрограмування алгоритмів множення цілих чисел у прямому та додатковому кодах у МОП на основі мікропроцесорних секцій ВС1 і ВУ4; навчитися розробляти імітаційні моделі МОП, проводити їхню симуляцію у середовищі Active – HDL та аналізувати отримані результати.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

Алгоритми множення чисел у форматі з фіксованою комою займають значне місце серед команд сучасних ЦОМ. Аналогічні алгоритми можуть також бути використовувані при виконанні операцій множення мантис чисел з рухомою комою.

### 6.1. Послідовність виконання лабораторної роботи

1. Для заданної операції множення цілих чисел розробити діаграму виконання операції.
2. Розробити функціональну схему мікрообчислювального пристрою (МОП), до складу якого входять БОД (на основі двох секцій ВС1), ВУ4, МПП, РМК, регістр станів на JK – тригерах. Припустити, що синхронізація усіх блоків МОП виконується одним тактовим сигналом CLK.
3. Розробити ГСМ та таблицю кодування (з коментарями) мікрооперацій БОД, ВУ4 та інших блоків МОП.
4*. Виконати налагодження VHDL – проекту МОП та провести симуляцію розробленої таблиці кодування (цей пункт роботи виконується факультативно).
5. Скласти звіт про виконану лабораторну роботу та захистити його.
6. Дати відповіді на контрольні запитання.

### 6.2. Варіанти індивідуальних завдань

Прийняти, що множене (Мн) "А" та множник (Мк) "В" задані у вигляді 8 – розрядних цілих чисел (старший розряд - знаковий). Конкретні значення варіантів індивідуальних завдань наведені у табл. 6.1 – табл. 6.4. При виконанні роботи розрядність БОД (дві секції ВС1) прийняти рівною розрядності початкових операндів, тобто вісім розрядів.

Спосіб множення. Таблиця 6.1

| (N)mod4 | Алгоритм множення | Аналіз розрядів Мк | Операція зсуву |
|---|---|---|---|
| 0 | "А" | З молодших | П (праворуч) |
| 1 | "Б" | З молодших | Мн (ліворуч) |
| 2 | "В" | З старших | П (ліворуч) |
| 3 | "Г" | З старших | Мн (праворуч) |

**Примітка.** П - часткові добутки.

Зображення операндів. Таблиця 6.2

| (N)mod2 | Представлення операндів |
|---|---|
| 0 | Прямий код |
| 1 | Додатковий код |

Розподіл регістрової пам'яті ВС1. Таблиця 6.3

| (N)mod3 | Множене | Множник |
|---|---|---|
| 0 | R1 | PQ |
| 1 | R10 | R6 |
| 2 | PQ | R13 |

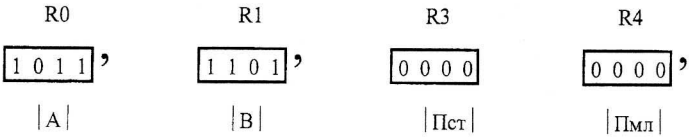
Операнди для контрольного прикладу. Таблиця 6.4

| (N)mod5 | Множене | Множник |
|---|---|---|
| 0 | - 5 | + 7 |
| 1 | + 6 | - 4 |
| 2 | - 10 | - 23 |
| 3 | + 4 | - 7 |
| 4 | - 5 | + 7 |

### 6.3. Методичні вказівки до виконання лабораторної роботи

#### 1.Множення модулів чисел

Як приклад, розглянемо алгоритм множення "А" (рис. 6.1). Нехай множене (Мн) $|A| = 11$, множник (Мк) $|B| = 13$. Виконаємо наступний розподіл регістрів РЗП ВС1:




У регістрах R3 і R4 будемо формувати старшу (Пст) та молодшу (Пмл) частини часткових добутків, а наприкінці множення - старшу та молодшу частину результату подвійної довжини (стосовно формату операндів).

З приклада рис. 6.1 випливає, що в процесі множення у формуванні часткових добутків беруть участь тільки старші чотири розряди Пст. Якщо одночасно на кожнім кроці виконувати зсув праворуч Пст та Пмл, то звільнюваний розряд множника можливо використати для розміщення спадаючих розрядів Пст (з цих розрядів буде поступово утворюватися Пмл). При використанні мікропроцесорної секції ВС1 це можливо здійснити в такий спосіб (рис. 6.2). По-перше, щоб запобігти втраті старшого розряду необхідно лінію вихідного переносу С4 підключити на вхід RF3 зсувача ЗСВГ.

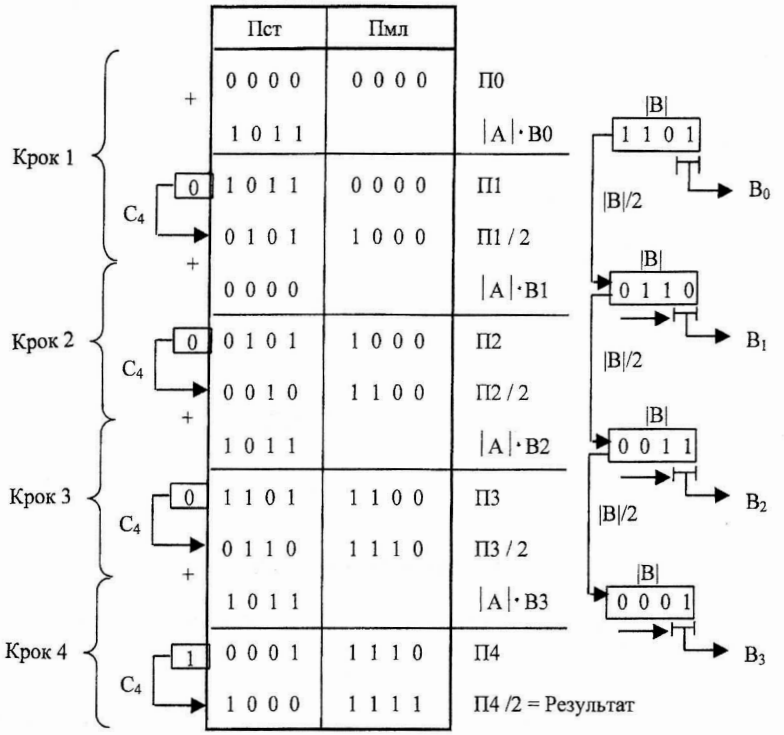



Рисунок 6.1. Діаграма множення модулів чисел по алгоритму "А"

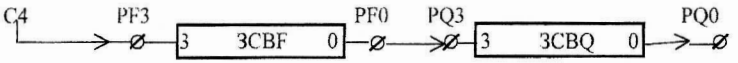



Рисунок 6.2. Організація операції зсуву на ВС1

По – друге, на кожнім кроці множення реалізується мікрооперація:

$$R3=\begin{cases} R3 + 0, \text{ коли } PQ0 = 0 \rightarrow I2\ I1\ I0 = 0\ 1\ 1; \\ R3 + R0, \text{ коли } PQ0 = 1 \rightarrow I2\ I1\ I0 = 0\ 0\ 1. \end{cases}$$

В зв'язку з цим для додавання джерел операндів А і В керуюча інструкція на вході АЛП ВС1 повинна мати вигляд (див. додаток Д1.1): I2 I1 I0 = 0 0 1 ∨ 0 1 1. Для автоматичного формування цих команд і часткових добутків необхідно подати на вхід I1 інструкції ВС1 значення PQ0 з виходу зсувача ЗСВQ. У цьому випадку в МПС необхідно виконати комутацію ліній, яка наведена на рис. 6.3.

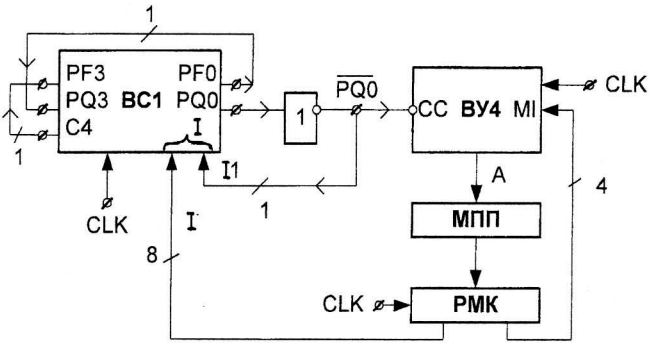



Рисунок 6.3. Функціональна схема МОП множення з алгоритмом "А"

### 2. Множення прямих кодів

Якщо операнди задані у прямому коді, то Апк = ЗнА |A|, Впк = ЗнВ |B|. Знак добутку формується як

$$\text{ЗнС} = \text{ЗнА} \oplus \text{ЗнВ}, \tag{6.2}$$

а модуль добутку обчислюється по заданому алгоритму окремо від знакового розряду у вигляді:

$$|C| = |A| \times |B|. \tag{6.3}$$

### 3. Множення додаткових кодів

При множенні у додатковому коді операнди розглядаються разом з своїми знаковими розрядами. Нехай А (Мн) = + 5, В (Мк) = - 3. Тоді маємо:

Адк = 0.1 0 1 → R1 ; Вдк = 1.1 0 1 → PQ ; Вдк = $\bar{1}$. 1 0 1 = - 8 + 4 + 1 = - 3.

При такому зображенні негативного множника на останньому кроці множення з використанням алгоритма "А" необхідно виконати корекцію часткового добутку (П – Адк). Якщо множник містить чотири розряди, то перші три кроки множення виконуються як беззнакові, а на останньому четвертому кроці - організується корекція (дивися діаграму рис. 6.4).

Нехай розподіл регістрів РЗП ВС1 має вигляд:

| R1 | PQ | R3 | PQ |
|---|---|---|---|
| 0.1 0 1 | 1.1 0 1 | 0.0 0 0 | 1.1 0 1 |
| Адк = Мн | Вдк = Мк | (Пст)дк | (Пмл)дк |

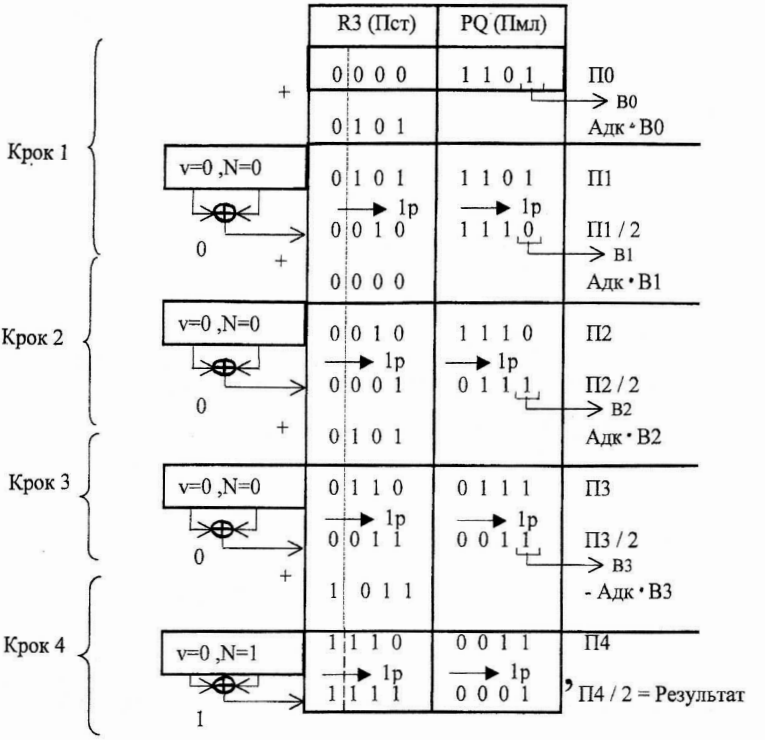



Рисунок 6.4. Діаграма множення додаткових кодів з використанням алгоритма "А"

На кожному кроці у операції підсумовування беруть участь тільки старші розряди часткових добутків Пст та множеного. Тому якщо одночасно виконувати зсув Пст і множника, то можливо організувати зсув подвійної довжини (рис. 6.5).

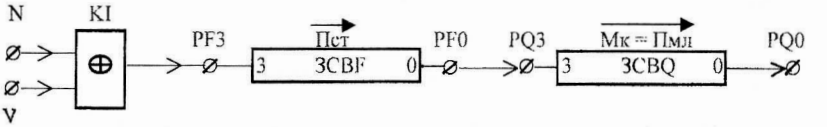



Рис. 6.5. Організація операції зсуву у додатковому коді

Керуючий інвертор КІ здійснює операцію над інформаційними сигналами переповнення V та знака N ВС1 згідно формули:

$$PF3 = V \oplus N = \overline{V} \cdot N \vee V \cdot \overline{N}, \qquad (6.4)$$

При цьому на вході PF3 зсувача ЗСВF формується правильне значення знака часткового добутку (знак суми N може бути зворотним в результаті виникнення переповнення АЛП V = 1).

Функціональна схема МОП наведена на рис. 6.6. На схемі: мультиплексор МХ відносно керуючого розряду S1 з РМК формує на вході I1 (перший розряд інструкції ВС1) код:

$$I1 = \begin{cases} \overline{PQ0}, & \text{коли } S1 = 0; \quad (n-1) \text{ - крок} \\ I1\,(\text{РМК}), & \text{коли } S1 = 1 \quad \text{(крок корекції).} \end{cases}$$

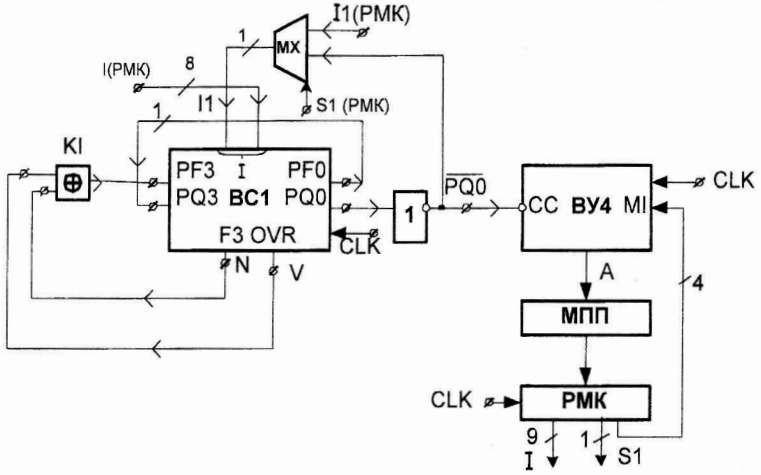



Рис. 6.6 Функціональна схема МОП множення додаткових кодів (алгоритм "А")

### 6.3. Зміст звіту

1. Діаграма виконання контрольного прикладу.
2. Функціональна схема МОП з короткими коментарями.
3. ГСМ та таблиця кодування.

4*. Листінг VHDL – проекту з результатами експериментальних досліджень (цей пункт роботи виконується факультативно).

### 6.4. Контрольні запитання

1. У якому випадку треба виконувати корекцію результату множення чисел в додатковому коді ?
2. Як організувати у 8 – розрядному МОП отримання 16 - розрядного часткового добутку при множенні з використанням алгоритма "Б" ?
3. Запропонуйте послідовність дій при виконанні операції зсуву 16 – розрядного часткового добутку ліворуч (алгоритм "В") на один розряд, якщо БОД містить вісім розрядів (дві секції ВС1).

**Лабораторна робота №7**

# РОЗРОБКА ТА ДОСЛІДЖЕННЯ ПРИСТРОЮ ДІЛЕННЯ ЦІЛИХ ЧИСЕЛ ОДИНИЧНОГО ТА ПОДВІЙНОГО ФОРМАТІВ

**МЕТА РОБОТИ:** придбати навички розробки та мікропрограмування алгоритмів ділення цілих чисел одиничного та подвійного форматів у прямому та додатковому кодах; навчитися розробляти імітаційни моделі МОП мовою VHDL та проводити їхню симуляцію у середовищі Active – HDL.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

Операція ділення відноситься до найбільш тривалих арифметичних операцій. Для визначення кожної цифри частки виконуються мікрооперації складання та зсуву залишку. Крім того, при діленні цілих чисел одиничного формату треба ще визначати кількість цифр у частки шляхом окремих циклів нормалізації діленого та дільника.

### 7.1. Послідовність виконання лабораторної роботи

Послідовність виконання цієї лабораторної роботи аналогічна послідовності виконання лабораторної роботи за № 6.

### 7.2. Варіанти індивідуальних завдань

Варіанти індивідуальних завдань наведені у табл. 7.1 – табл. 7.4. Передбачається, що мікропрограму ділення необхідно оформити у вигляді підпрограми та забезпечити її виклик з основної програми.

Алгоритми ділення. Таблиця 7.1

| (N)mod4 | Алгоритм ділення | Формат операндів | Операція зсуву залишків |
|---|---|---|---|
| 0 | "а" | n / n | Ліворуч |
| 1 | "а" | 2 n / n | Ліворуч |
| 2 | "б" | n / n | Праворуч |
| 3 | "б" | 2 n / n | Праворуч |

Зображення операндів. Таблиця 7.2

| (N)mod2 | Представлення операндів |
|---|---|
| 1 | Прямий код |
| 0 | Додатковий код |

Розподіл регістрової пам'яті BC1. Таблиця 7.3

| (N)mod3 | Ділене | Дільник |
|---|---|---|
| 0 | R5 | PQ |
| 1 | R12 | R7 |
| 2 | PQ | R15 |

Операнди для контрольного прикладу. Таблиця 7.4

| (N)mod5 | Ділене | Дільник |
|---|---|---|
| 0 | ± 76 | + 5 |
| 1 | ± 58 | - 4 |
| 2 | ± 102 | - 13 |
| 3 | ± 99 | - 9 |
| 4 | ± 68 | + 7 |

## 7.3. Методичні вказівки до виконання лабораторної роботи

### 1. Алгоритм ділення одиничного формату $A^n / B^n$

Ділене (Дм) Апк і дільник (Дк) Впк у прямому коді складаються із знаку та модулю:

$$Апк = ЗнА \cdot |A|'; \qquad Впк = ЗнВ \cdot |B|'. \tag{7.1}$$

При використанні чисел одиничного формату мають місце співвідношення:

$$ЗнD = ЗнA \oplus ЗнB; \tag{7.2}$$

$$A' / B' = D' + C' / B' = F', \tag{7.3}$$

де ЗнD - знак частки; D' - модуль частки; C' - залишок операції ділення.

Особливістю алгоритму ділення чисел одиничного формату є те, що заздалегідь не відомо кількість розрядів частки. Для визначення довжини розрядів частки $L_D$ використовується співвідношення:

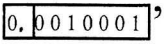
$$L_D = P_B^H - P_A^H + 1, \tag{7.4}$$

де $P_B^H$ і $P_A^H$ - кількість зсувів B' та A' при нормалізації відповідно B' та A'.

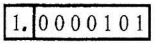
В деяких випадках операція ділення блокується. По – перше, якщо B' = 0, фіксується переповнення і ділення не виконується. По – друге, якщо A' < B', ділення також не виконується. При цьому приймається D' = 0 і C' = A'.

Розглянемо приклад ділення чисел A = +17 і B = -5, якщо n = 8. В одиничному форматі ці числа мають наступне зображення:

$A^n_{пк}$

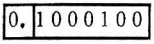



$B^n_{пк}$



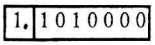

Після нормалізації A' і B' (шляхом зсуву ліворуч A' і B'), одержуємо:

$ЗнA \cdot | A_н |'$




← 2р. ($Р^н_а = 2$)

$ЗнB \cdot | B_н |'$




← 4р. ($Р^н_в = 4$)

Таким чином, для довжини частки одержимо $L_D = 4 - 2 + 1 = 3$. Діаграма ділення нормалізованих модулів $|A_н|'$ та $|B_н|'$ для визначення розрядів D' ($L_D = 3$) з використанням алгоритма "а" наведена на рис. 7.1.

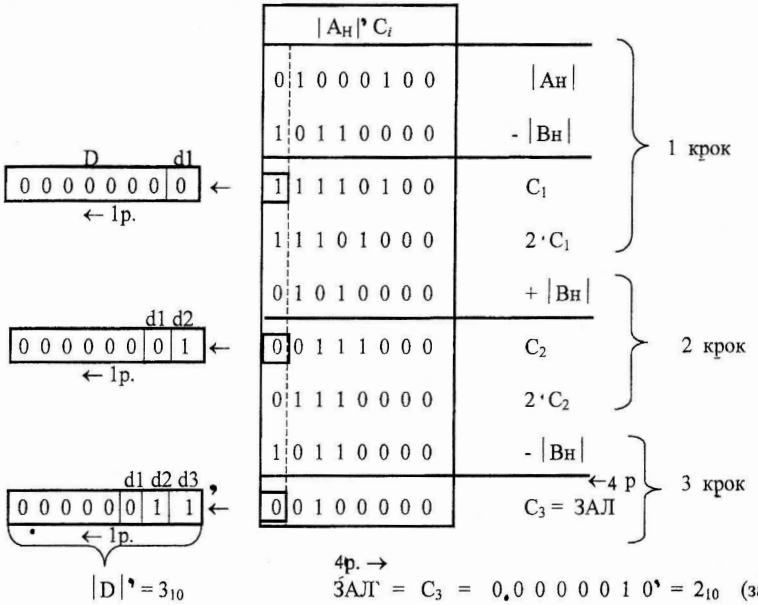
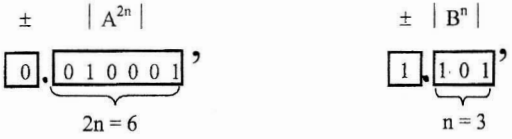



Рисунок 7.1. Діаграма ділення модулів чисел одиничного формату

В десятковій системі для заданих чисел, очевидно, маємо 17 / 5 = 3 (2), що підтверджує коректність двійкового результату.

### 2. Алгоритм ділення подвійного формату $A^{2n}$ / $B^n$

Для спрощення алгоритму ділення необхідно для діленого (A) у ЦОМ призначити два регістри, а для представлення дільника (B) – один. Наприклад, при A = + 17, B = - 5 і n = 3 маємо:




Таким чином, максимальна кількість значащих цифр (довжина) частки у цьому випадку складає:

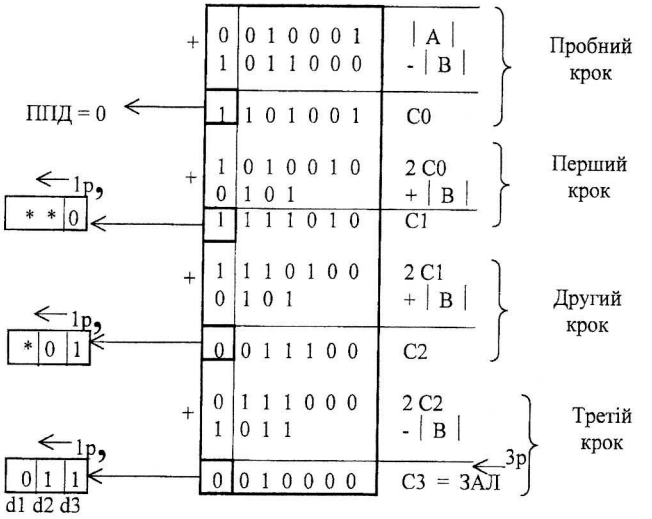
$$L_D^{max} = L_A - L_B + 1 = 2n - n + 1 = n + 1. \quad (7.5)$$

Якщо частка D' розміщується в n розрядах і D'(n + 1) =1, виникає переповнення розрядної сітки D', тобто при

$$C_1 = |A^{2n}| - |B^n| \geq 0 \qquad (7.6)$$

операція ділення блокується. У цьому випадку зводиться прапор переповнення (ППД = 1) і операція ділення на цьому завершується. У протилежному випадку лічильник циклів ділення ЛЧ установлюється в стан n і починається цикл ділення. Операція завершується за нульовим значенням лічильника.

Приклад ділення подвійного формату на одиничний наведено на рис. 7.2.




3р. →

ЗАЛ = $C_3$ = 0,0 0 0 0 1 0' = $2_{10}$ (залишок)

Рисунок 7.2. Діаграма ділення модулів чисел подвійного формату

В десятковій системі для заданих чисел, очевидно, маємо 17 / 5 = 3 (2), що підтверджує коректність двійкового результату.

### 7.4. Зміст звіту

Пункти звіту цієї лабораторної роботи відповідні змісту звіту до лабораторной роботи за № 6.

### 7.5. Контрольні запитання

1. Перелічить дії, які необхідно виконати для визначення кількості цифр частки в операції ділення чисел одиничного формату.
2. У якому разі та чому неможливо виконати ділення чисел подвійного формату ?
3. Яким чином визначається поточна цифра частки в алгоритмах ділення прямих та додаткових кодів ?

## Лабораторна робота №8

# ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ МІКРОПРОГРАМУЮЧИХ ПРИСТРОЇВ НА БАЗІ СЕКЦІЙ ВУ1 І ВС2

**МЕТА РОБОТИ:** опанувати принципи побудови та функціонування мікропроцесорних секцій керування адресою мікрокоманди (СКАМ) К1804ВУ1 та блоків обробки даних (БОД) К1804ВС2; придбати навички розробки і кодування мікроалгоритмів найпростіших функцій; навчитися розробляти імітаційни моделі блоків на основі секцій ВУ1 та ВС2, проводити їхню симуляцію та аналізувати отримані результати.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

Мікропроцесорна секція ВУ1 призначена для побудови блоків мікропрограмного керування. На відміну від мікросхеми ВУ4, вона має чотири розряди і може формувати адреси 16 осередків мікропрограмної пам'яті БМК. При необхідності адресації більшого числа осередків МПП мікросхеми ВУ1 можуть бути об'єднані між собою (за допомогою спеціальних сигнальних клем) і утворити 8, 12, 16, ... розрядні формувачі адреса.

Мікропроцесорна секція ВС2 призначена для побудови блоків обробки данных (БОД) з розрядністю слова, яка також кратна чотирьом. На відміну від аналогічної секції ВС1, секція ВС2 має більш повний набір інструкцій арифметико – логічного блоку. Крім того, до її складу входить блок побудови спеціальних функцій, використання яких полегшує реалізувати операції множення, ділення, нормалізації (і деякі інші операції) над модулями чисел або над числами у додатковому коді.

Більш докладний опис секцій ВУ1 та ВС2 приведено у додатку Д1.3 і Д2.2.

### 8.1. Послідовність виконання лабораторної роботи

1. Вивчити з використанням літературних джерел [1 – 5] та дійсних методичних вказівок склад, призначення та команди секцій ВУ1 та ВС2.
2. Для заданого алгоритму розробити ГСМ та таблицю кодування у вигляді основної програми та підпрограми. У ГСМ забезпечити введення в регістрову пам'ять ВС2 операндів з використанням зовнішньої шини DA (або DB).

3*. Розробити VHDL – модель мікрообчислювального пристрою на основі ВУ1, ВС2, МПП, РМК та регістра станів на основі JK – тригерів з керованою синхронізацією.

4*. Виконати налагодження VHDL – проекту та провести симуляцію роботи МОП.

5. Скласти звіт з виконання лабораторної роботи та захистити його.
6. Дати відповіді на контрольні запитання.

**Примітка.** Пункти 3* та 4* виконуються факультативно.

### 8.2. Варіанти індивідуальних завдань

Варіанти індивідуальних завдань наведені у табл. 8.1. У таблиці наведено алгоритм операції (множення або ділення) та спосіб зображення алгебраічних чисел (прямий або додатковий). Варіант вибирається з використанням номера N за модулем 12.

Варіанти індивідуальних завдань. Таблиця 8.1

| (N)mod12 | Алгоритм операції | Код операндів |
|---|---|---|
| 0 | Множення "А" | Прямий |
| 1 | Множення "Б" | Прямий |
| 2 | Множення "В" | Прямий |
| 3 | Множення "Г" | Прямий |
| 4 | Множення "А" | Додатковий |
| 5 | Множення "Б" | Додатковий |
| 6 | Множення "В" | Додатковий |
| 7 | Множення "Г" | Додатковий |
| 8 | Ділення "а" | Прямий |
| 9 | Ділення "б" | Прямий |
| 10 | Ділення "а" | Додатковий |
| 11 | Ділення "б" | Додатковий |

### 8.3. Методичні вказівки до виконання лабораторної роботи

Функціональна схема мікрообчислювального пристрою (МОП) навсдена на рис. 8.1. До її складу входять схема керування адресою мікрокоманди (СКАМ) на базі мікросхеми ВУ1, БОД на основі секцій ВС2, мікропрограмна пам'ять (МПП), регістр мікрокоманд (РМК), три мультиплексора МХ1 – МХ3, генератор G тактових сигналів і тригер для керування режимами запуску та зупинки роботи МОП. Припустимо також, що у системі використана однофазна система синхронізації.

При натисканні на кнопку "Pusk" на пульті керування формується одиничний короткочасовий імпульс низького рівня, по якому тригер керування зводиться в одиничний стан. Після цього дозволяється подача тактових сигналів CLK в БОД, СКАМ і РМК. Одночасно імпульс від кнопки "Pusk" надходить на клему $\overline{ZA}$ ВУ1. В результаті на виході Y(3 –0) СКАМ формується нульова адреса першої мікрокоманди. З МПП в РМК записується мікрокоманда (МК). РМК вміщує спеціальні поля керування блоками СКАМ, БОД і трьома мультиплексорами. У залежності від вмісту РМК S2, S3 та A1, та ознак C4 і Z БОДа на виводах SE0 і SE1 мультиплексорів формуються сигнали (рис. 8.2), які далі надходять на входи S1 і S0 мікросхеми ВУ1. Далі ці сигнали формують адресу наступної мікрокоманди (див. додаток Д2.2).

**MX1**

| S2 | S3 | CC |
|---|---|---|
| 0 | 0 | "0" |
| 0 | 1 | "1" |
| 1 | 0 | C4 |
| 1 | 1 | Z |

**MX2**

| A1 | CC | SE1 | |
|---|---|---|---|
| 0 | 0 | 0 | ЛМК |
| 0 | 1 | 0 | РА |
| 1 | 0 | 1 | ST |
| 1 | 1 | 1 | DA |

**MX3**

| A1 | CC | SE0 |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Рисунок 8.2. Логіка керування мультиплексорами МХ1 - МХ3

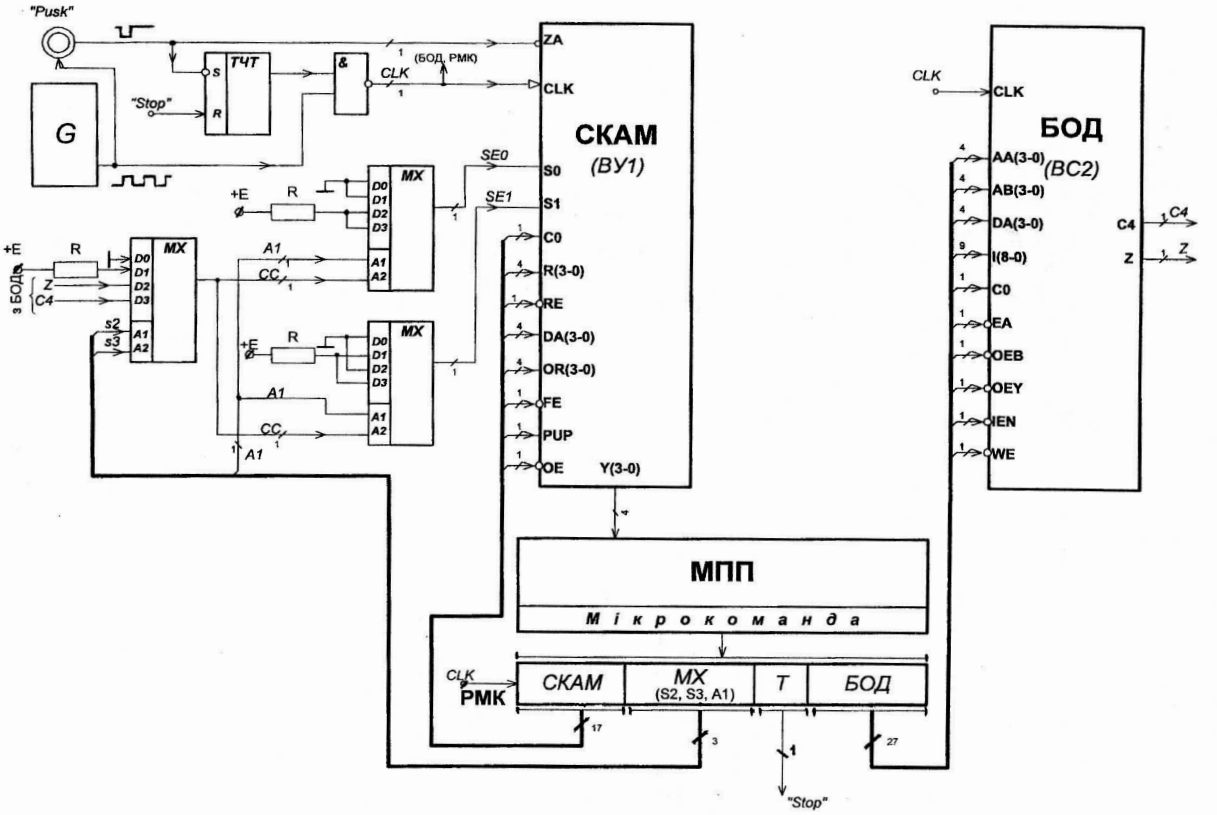



**Рисунок 8.1.** *Функціональна схема мікрообчислювального пристрою*

У форматі мікрокоманди є також поле "T" (Stop), за допомогою якого формується керуючий сигнал "Stop" високого рівня. Цей сигнал скидає тригер керування в нульовий стан. При цьому здійснюється заборона формування тактових сигналів і МОП зупиняє свою роботу.

Як приклад розглянемо ГСМ, наведену на рис. 8.3. Вона складається з основної програми (починається з нульової адреси) і підпрограми ПП1 (починається з п'ятої адреси).

Виклик ПП1 здійснюється з основної програми мікрокомандою з нульовою адресою. У підпрограмі здійснюється два цикли: один цикл інкрементування вмісту R3 БОД доти, поки сигнал вихідного переносу C4 не прийме одиничне значення. Після цього в R3 завантажується код 1111 і здійснюється цикл декрементування вмісту R3 доти, поки ознака нульового результату Z на виході БОД не стане рівною одиниці. У цьому випадку відбудеться вихід з підпрограми в основну програму, де буде сформований сигнал "Stop" та МОП завершить свою роботу.

Таблиця кодування розглянутого фрагмента ГСМ наведена в табл. 8.3 і табл. 8.4.

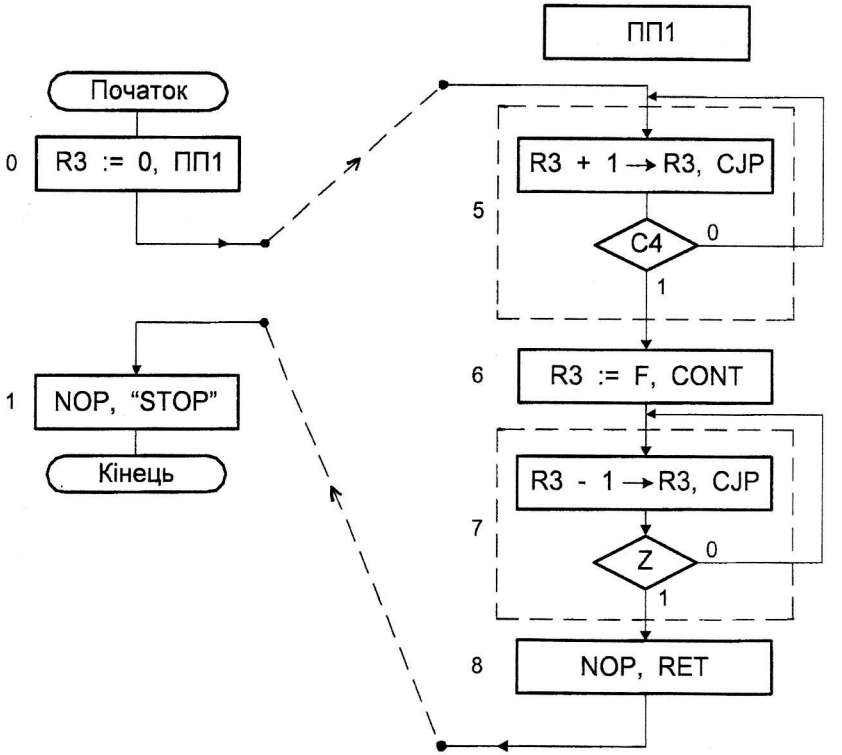



Рисунок 8.3. Приклад ГСМ

Таблиця кодування СКАМ ВУ1. Таблиця 8.3

| Адреса МК | R | DA | OR | C0 | $\overline{FE}$ | PUP | $\overline{RE}$ | $\overline{OE}$ | Примітки |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 5 | 5 | 0 | 1 | 0 | 1 | 0 | 0 | Push, виклик ПП1, 5 → РА |
| 1 | * | * | 0 | 1 | 1 | * | 1 | 0 | Stop |
| --- | -- | -- | -- | -- | -- | -- | -- | -- | ---------- |
| 5 | * | * | 0 | 1 | 1 | * | 1 | 0 | Умовний перехід CJP (ЛМК, РА) |
| 6 | 7 | * | 0 | 1 | 1 | * | 0 | 0 | CONT, 7 → РА |
| 7 | * | * | 0 | 1 | 1 | * | 1 | 0 | Умовний перехід CJP (ЛМК, РА) |
| 8 | * | 0 | 0 | 1 | 0 | 0 | 1 | 0 | Pop, вихід з підпрограми ПП1 |

Таблиця кодування БОД (ВС2), МХ1 – МХ3, stop. Таблиця 8.4

| Адреса МК | $\overline{EA}$ | I0 | $\overline{OEB}$ | I(4-1) | I(8-5) | AA | AB | C0 | S2 S3 A1 | Stop | Примітки |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 8 | 4 | 3 | 3 | 0 | 0 1 1 | 0 | 0 → R3 |
| 1 | 0 | 0 | 0 | 8 | 5 | 3 | 3 | 0 | 0 0 0 | 1 | NOP, Stop |
| -- | -- | -- | -- | -- | -- | -- | -- | -- | --------- | - | ---------- |
| 5 | 0 | 0 | 0 | 4 | 4 | 0 | 3 | 1 | 1 0 0 | 0 | R3 + 1 → R3 |
| 6 | 0 | 0 | 0 | 0 | 4 | 3 | 3 | 0 | 0 0 0 | 0 | F → R3 |
| 7 | 0 | 0 | 0 | 5 | 5 | 3 | 3 | 1 | 1 1 0 | 0 | R3 + 1 → R3 |
| 8 | 0 | 0 | 0 | 8 | 5 | 3 | 3 | 0 | 0 0 1 | 0 | NOP |

### 8.4. Зміст звіту

1. Функціональна схема МОП (з поясненнями назви її блоків та сигналів).
2. ГСМ і таблиця кодування роботи МОП з коментарями.
3. Приклад виконання операції (числові дані прийняти самостійно).
4*. Листінг VHDL - проекту з результатами експериментальних досліджень (цей пункт виконується факультативно).

### 8.5. Контрольні запитання

1. Якім чином організувати спільну роботу в БМК мікросхеми ВУ1 та кодового перетворювача сигналів К1804ВУ3 ? Наведіть пропонуєму функціональну схему.
2. Як з використанням спеціальних функцій ВС2 організувати:
   А) - множення додаткових кодів ?
   Б) - ділення модулів ?
3. Якім чином розрахувати кількість секцій ВУ1, коли організація МПП складає 2К комірок по 64 розряди кожна ?

**Лабораторна робота №9**

# ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ КОНТРОЛЮЮЧОГО ПРИСТРОЮ ОПЕРАЦІЇ МНОЖЕННЯ ЧИСЕЛ

**МЕТА РОБОТИ:** придбати навички розробки та кодування алгоритмів контролю множення цілих чисел у прямому та додатковому кодах з використанням залишків по модулю три (mod 3) та по модулю сім (mod 7); навчитися розробляти VHDL – моделі цих алгоритмів та проводити їхню симуляцію у середовищі Active – HDL.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

Якщо у якості контрольного коду використовується залишок за mod 3 (або mod 7), то у якості контрольній операції над залишками може бути обрана та ж сама операція множення, яка виконується над початковими числами: множеним "А" і множником "В":

$$R^{m3}(A \times B) = R^{m3}[\,R^{m3}(A) \times R^{m3}(B)\,], \qquad (9.1)$$

де $R^{m3}(A)$ - позначає залишок числа А за модулем три.

Обчислювальний алгоритм контролю операції множення зводиться до наступних послідовних кроків:

1. Для множеного А и множника В знаходяться контрольні залишки R(A) і R(B) за заданим модулем (три або сім).
2. Знаходиться добуток операндів: $C = A \times B$.
3. Обчислюється контрольний код від отриманого добутку: $r = R(C)$.
4. Знаходиться добуток над контрольними залишками та визначається його контрольний код: $R\,[\,R(A) \times R(B)\,]$.
5. Виконується порівняння контрольного коду основної операції r з результатом операції над контрольними кодами R. При розбіжності результату формується сигнал помилки: ERROR = 1.

### 9.1. Послідовність виконання лабораторної роботи

1. Для заданної операції контролю множення цілих чисел розробити діаграму виконання контрольного прикладу.

2. Розробити ГСМ та таблицю кодування (з коментарями всіх операцій) в МОП на основі двох секцій ВС1, ВУ4, МПП, РМК.

3*. Виконати налагодження VHDL – проекту алгоритмів контролю та провести симуляцію моделі пристрою, яка була збудована у лабораторній роботі за № 6 (цей пункт роботи виконується факультативно).

4. Скласти звіт з виконання лабораторної роботи та захистити його.

5. Дати відповіді на контрольні запитання.

### 9.2. Варіанти індивідуальних завдань

Алгоритм множення, код зображення операндів, розташування операндів та початкові числа для контрольного прикладу треба взяти з табл. 9.1 – табл. 9.4. Код використання залишків наведено у табл. 9.5.

Спосіб множення. Таблиця 9.1

| (N)mod4 | Алгоритм множення | Аналіз розрядів Мк | Операція зсуву |
|---|---|---|---|
| 0 | "А" | З молодших | П (праворуч) |
| 1 | "Б" | З молодших | Мн (ліворуч) |
| 2 | "В" | З старших | П (ліворуч) |
| 3 | "Г" | З старших | Мн (праворуч) |

**Примітка.** П - часткові добутки.

Зображення операндів. Таблиця 9.2

| (N)mod2 | Представлення операндів |
|---|---|
| 0 | Прямий код |
| 1 | Додатковий код |

Розподіл регістрової пам'яті ВС1. Таблиця 9.3

| (N)mod3 | Множене | Множник |
|---|---|---|
| 0 | R1 | PQ |
| 1 | R10 | R6 |
| 2 | PQ | R13 |

Операнди для контрольного прикладу. Таблиця 9.4

| (N)mod5 | Множене | Множник |
|---|---|---|
| 0 | - 5 | + 7 |
| 1 | + 6 | - 4 |
| 2 | - 10 | - 23 |
| 3 | + 4 | - 7 |
| 4 | - 5 | + 7 |

Зображення коду залишків. Таблиця 9.5

| (N)mod2 | Контрольний код |
|---|---|
| 0 | Mod 3 |
| 1 | Mod 7 |

### 9.3. Методичні вказівки до виконання лабораторної роботи

Контроль операції множення здійснюється за допомогою контрольних кодів, які являють собою згорнення початкових операндів (множеного A і множника B) та результату C за модулем три ($R^{m3}$) або за модулем сім ($R^{m7}$) [5].

## 1. Контроль у прямому коді

Коли множене А та множник В задані у прямому коді, можливо записати:

$$A_{ПК} = ЗнА \cdot A'; \qquad B_{ПК} = ЗнВ \cdot B', \tag{9.2}$$

де $A' = |A|$ - модуль числа А; $B' = |B|$ - модуль числа В;

Зн А, Зн В - знакові розряди.

Нехай наприклад, $A = -11_{10}$, $B = +13_{10}$.

Тоді $A_{ПК} = 1.1011'$ ; $B_{ПК} = 0.1101'$ ;

$A' = |A| = .1011'$ ; $B' = |B| = .1101'$ .

Контрольні коди А' та В' за модулем три (mod 3) мають вигляд:

$$R^{m3}_{A'} = R^{m3}(\underbrace{10 \quad 11}) = (10) = 2_{10}; \tag{9.3}$$
$$R^{m3}_{B'} = R^{m3}(\underbrace{11 \quad 01}) = (01) = 1_{10};$$

Функціональна схема блока контролю наведена на рис. 9.1. В арифметичному пристрої множення модулів чисел (АПМ) відбувається операція C'= A'·B' по одному із відомих алгоритмів (наприклад, А, Б, В або Г): (1011') x (1011') = ( 1000 1111').

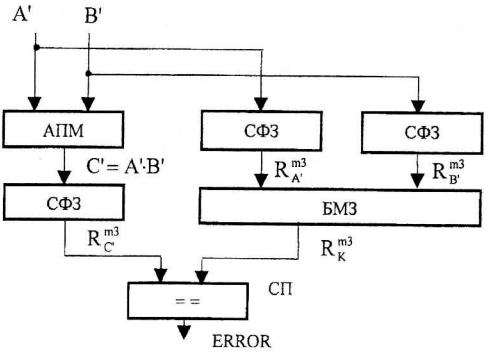



Рисунок 9.1. Функціональна схема блоку множення з контролем операції за модулем три

Схема згорнення залишку (СФЗ) формує слідуючий результат:

$$R^{m3}_{C'} = R^{m3}(\underbrace{10 \quad 00 \quad 11 \quad 11}) = (10) = 2_{10}. \tag{9.4}$$

Блок множення залишків (БМЗ) виконує контрольну операцію множення над залишками операндів:

$$R^{m3}_{K} = R^{m3}_{A'} \times R^{m3}_{B'} = (10) \times (01) = (10) = 2_{10} \tag{9.5}$$

відносно табл. 9.6 На основі цієї таблиці складаються карти Карно та знаходяться значення розрядів К2 і К1 контрольного коду.

Таблиця істиності БМЗ. Таблиця 9.6

| $R^{m3}_{A'}$ | | $R^{m3}_{B'}$ | | $R^{m3}_{K}$ | |
|---|---|---|---|---|---|
| X2 | X1 | Y2 | Y1 | K2 | K1 |
| 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 |

| X2·X1 \ Y2·Y1 | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | * | * | * | * |
| 01 | * | 0 | 1 | 1 |
| 11 | * | 1 | 1 | 1 |
| 10 | * | 1 | 1 | 0 |

$K2 = Y2 \cdot X1 \vee Y1 \cdot X2$

| X2·X1 \ Y2·Y1 | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | * | * | * | * |
| 01 | * | 1 | 1 | 0 |
| 11 | * | 1 | 1 | 1 |
| 10 | * | 0 | 1 | 1 |

$K1 = Y1 \cdot X1 \vee Y2 \cdot X2$

Схема порівняння (СП) на рис. 9.1 виконує операцію аналіза залишків:

$$(R^{m3}_{C'}) == (R^{m3}_{K}). \qquad (9.6)$$

В випадку незбігу формується сигнал помилки (ERROR=1). При цьому будуть виявлені усі одинокі помилки, а також деяка частина подвійних помилок, при яких правильний і помилковий результат множення чисел мають незбіжні залишки за модулем три.

## 2. Контроль додаткових кодів

При виконанні множення у додатковому коді знак числа також приймає участь в операції множення нарівні з цифровими розрядами. Як відомо, якщо число А позитивне, можливо записати:

$$A_{дк} = A_{пк} = ЗнА \cdot A'. \qquad (9.7)$$

У випадку, якщо $A < 0$, маємо:

$$A_{дк} = 2^n - A', \qquad (9.8)$$

де $2^n$ - вага $(n+1)$ – го розряду (див. рис. 9.2, а).

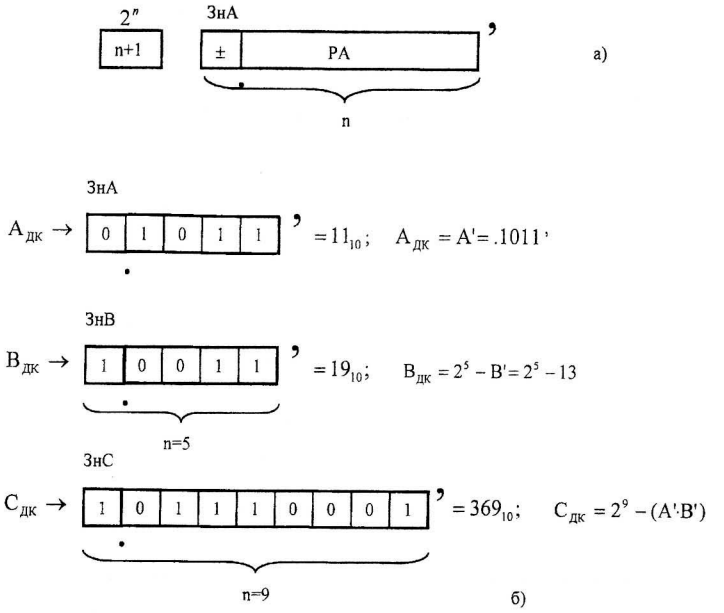



Рисунок 9.2. Зображення початкових даних та результату множення у додатковому коді

Нехай, наприклад, множене $A = +11_{10}$; множник $B = -13_{10}$, тобто

$$A_{дк} = 0.1011'; \quad R^{m3}_{A_{дк}} = R^{m3}(\underbrace{00}.\underbrace{10}.\underbrace{11}) = (10) = 2_{10};$$
$$B_{дк} = 1.0011'; \quad R^{m3}_{B_{дк}} = R^{m3}(\underbrace{01}.\underbrace{00}.\underbrace{11}) = (01) = 1_{10};$$

Після виконання операції множення операндів по одному із алгоритмів (А, Б, В або Г) одержимо:

$$C_{дк} = A_{дк} \times B_{дк}. \tag{9.9}$$

Формат Сдк наведено на рис. 9.2, б. При цьому можливо записати наступні вирази:

$$B' = 2^5 - B_{дк};$$
$$C_{дк} = 2^9 - (A' \cdot B') = 2^9 - [A' \cdot (2^5 - B_{дк})] = 2^9 - 2^5 \cdot A' + A' \cdot B_{дк}. \tag{9.10}$$

Очевидно, контрольні коди за mod3 обох частин виразу (9.10) мають вигляд:

$$R_{C_{ДК}} = R_{2^9} - R_{2^5} \cdot R_{A'} + R_{A'} \cdot R_{B_{ДК}} . \qquad (9.11)$$

У загальному випадку отримаємо наступну формулу:

$$R_{C_{ДК}} = R_{\delta 2n} - R_{\delta n+1} \cdot R_{A'} + R_{A'} \cdot R_{B_{ДК}}, \qquad (9.12)$$

де $\delta 2n$ - вага 2n-го розряду;
$\delta n+1$ - вага (n+1)-го розряду.

Для наведеного прикладу маємо:

$$R_{C_{ДК}} = R^{m3}(\underbrace{01}\,\underbrace{01}\,\underbrace{11}\,\underbrace{00}\,\underbrace{01}) = (11) = 3_{10};$$
$$R_{A'} = R^{m3}(\underbrace{00}\,\underbrace{10}\,\underbrace{11}) = (10) = 2_{10};$$
$$R_{B_{ДК}} = R^{m3}(\underbrace{01}\,\underbrace{00}\,\underbrace{11}) = (01) = 1_{10};$$
$$R_{\delta 2n} = (10) = 2_{10};\ R_{\delta n+1} = (10) = 2_{10}.$$

Таким чином,

$$R_K = R_{\delta 2n} - R_{\delta n+1} \cdot R_{A'} + R_{A'} \cdot R_{B_{ДК}} = (10) - (10 \times 10) + (10 \times 01) = (11) = 3_{10};$$
$$R_K == R_{C_{ДК}}.$$

У тому випадку, якщо використовується контроль за mod7, отримаємо наступні вирази:

$$R_{C_{ДК}} = R^{m7}(\underbrace{1}\,\underbrace{01}\,\underbrace{110}\,\underbrace{001}) = (101) = 5_{10};$$
$$R_{A'} = R^{m7}(\underbrace{00}\,\underbrace{1}\,\underbrace{011}) = (100) = 4_{10};$$
$$R_{B_{ДК}} = R^{m3}(\underbrace{01}\,\underbrace{0}\,\underbrace{011}) = (101) = 5_{10};$$
$$R_{\delta 2n} = (001) = 1_{10};\quad R_{\delta n+1} = (100) = 4_{10}.$$

$$R_K = (001) - (100) \times (100) + (100) \times (101) = (001) - (010) + (110) = (101) = 5_{10};$$
$$R_K == R_{C_{ДК}}.$$

**Зміст звіту**

1. Діаграма виконання контрольного прикладу.
2. ГСА та таблиця кодування з коментарями.
3*. Результати симуляції експериментальних досліджень VHDL – проекту (цей пункт роботи виконується факультативно).

### 9.4. Контрольні запитання

1. Поясніть, які типи помилок дозволяє виявити контрольний код операції множення:
   а) - mod 3; б) - mod 7.
2. Як з використанням процесорних секцій ВС1 та секції ВУ4 організувати визначення залишку числа в додатковому коді за модулем три (або за модулем сім)?

## Лабораторна робота № 10

# РОЗРОБКА ТА СИМУЛЯЦІЯ АРИФМЕТИЧНИХ ПРИСТРОЇВ (АП) ДЛЯ СКЛАДАННЯ ТА ВІДНІМАННЯ ДАНИХ У ФОРМАТІ З РУХОМОЮ КОМОЮ

**МЕТА РОБОТИ:** вивчення алгоритмів складання та віднімання чисел у форматі з рухомою комою і способів їх виконання та симуляції у секційних мікропроцесорах серії К1804.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

Алгоритм складання (віднімання) дійсних чисел у ЦОМ містить декілька етапів. При цьому виконання операції віднімання відрізняється від складання тим, що на початку операції знак другого числа змінюється на зворотний. У цьому разі послідовність етапів виконання операції складання (або віднімання) полягає в наступному:

1. Виробляється порівняння порядків шляхом віднімання з порядку числа А (Ра) порядку числа В (Рв). Якщо порядки не рівні (Ра ≠ Рв), то переходять до процедури вирівнювання порядків. При цьому порядок меншого по модулю числа приймають рівним порядку більшого числа, а мантиса меншого числа зсувається праворуч на кількість S – ричних розрядів, яке дорівнює різниці порядків чисел (S – підстава характеристики). Частина розрядів переутвореної мантиси при зсуванні може губитися. Це означає, що дії над дійсними числами не є точними.
2. Виконується складання (віднімання) мантис чисел за правилами роботи ЦОМ з фіксованої комою.
3. Виконується аналіз нормалізації отриманого результату (мантиса результату повинна мати вигляд правильного дробу) і в разі потреби отримана сума (різниця) нормалізується.
4. При виконанні операції може виникнути переповнення розрядної сітки порядку. При цьому у ЦОМ формується сигнал переповнення порядку (FV = 1).

### 10.1. Послідовність виконання лабораторної роботи

1. Вивчити з використанням літературних джерел [1, 2] і методичних вказівок етапи та приклади виконання операцій складання та віднімання чисел у ЦОМ з рухомою комою.
2. Виконати розрахунок розрядної сітки і розробити функціональну схему процесора для виконання заданої операції.
3. Привести приклад виконання типових операцій у заданому форматі (по окремим крокам) з коментарями.
4. Разробити схему алгоритму виконання операції на рівні мікрооперацій у регістрах процесора. Алгоритм представити у вигляді основної частини та окремих процедур: порівняння порядків, вирівнювання порядків, складання (віднімання) мантис, нормалізація результату та виявлення переповнення розрядної сітки.
5. Розробити таблицю кодування основної частини алгоритму та окремих процедур, виконати опис мікропрограми та окремих полів мікрокоманди.
6.* Виконати налагодження VHDL - проекту на основі моделі мікрообчислювального пристрою (МОП) з лабораторної роботи №3. Провести симуляцію роботи МОП в середовищі **Active – HDL**.
7. Розрахувати тривалість одного такту роботи процесора і виконання всієї операції в цілому (з обліком "гіршого" та "кращого" випадків).

8. Скласти звіт з виконаної лабораторної роботи та захистити його.
9. Дати відповіді на контрольні запитання.

**Примітка.** Пункти, які відзначені знаком (*) виконуються факультативно.

### 10.2 Варіанти індивідуальних завдань

Варіанти індивідуальних завдань наведені у табл.10.1 – табл.10.8. Тип заданої процедури, яка підлягає розробці, приведений в табл.10.8.

Операція АП з рухомою комою. Таблиця 10.1

| (N)mod2 | Операція |
|---|---|
| 0 | Складання |
| 1 | Віднімання |

Базові мікросхеми АП. Таблиця 10.2

| (N)mod2 | Основні інтегральні схеми |
|---|---|
| 0 | К1804ВС1, К1804ВУ1 , К555КП12 , К1804ИР1 |
| 1 | К1804ВС2, К1804ВУ4 , К555КП12 , К1804ИР1 |

Діапазон чисел АП. Таблиця 10.3

| (N)mod4 | Діапазон чисел АП |
|---|---|
| 0 | $10^{\pm 18}$ |
| 1 | $10^{\pm 10}$ |
| 2 | $10^{\pm 30}$ |
| 3 | $10^{\pm 76}$ |

Точність зображення чисел. Таблиця 10.4

| (N)mod4 | Точність зображення мантис чисел |
|---|---|
| 0 | 7 десяткових цифр |
| 1 | 10 десяткових цифр |
| 2 | 12 десяткових цифр |
| 3 | 16 десяткових цифр |

Код для зображення порядку і мантиси. Таблиця 10.5

| (N)mod4 | Представлення порядку | Представлення мантис |
|---|---|---|
| 0 | Додатковий код (ДК) | Прямий код |
| 1 | З позитивним нулем (ПН) | Прямий код |
| 2 | З негативним нулем (НН) | Прямий код |
| 3 | З позитивним нулем (НН) | Прямий код |

Підстава характеристики. Таблиця 10.6

| (N)mod4 | Підстава характеристики |
|---|---|
| 0 | 2 |
| 1 | 4 |
| 2 | 8 |
| 3 | 16 |

Операнди для виконання контрольного приклада. Таблиця 10.7

| (N)mod4 | Ісходні числа для виконання приклада |
|---|---|
| 0 | A = + 32 , 125 ; B = - 10 , 725 |
| 1 | A = - 105 , 25 ; B = + 12 , 625 |
| 2 | A = + 0 , 0125 ; B = - 5 , 25 |
| 3 | A = - 27 , 125 ; B = - 7 , 5 |

Тип процедури, яка самостійно розробляється в проекте. Таблиця 10.8

| (N)mod3 | Тип заданої процедури |
|---|---|
| 0 | Вирівнювання порядків |
| 1 | Складання мантис |
| 2 | Нормалізація результату |

**Примітка.** При розробці мікропрограми і таблиці кодування процедур треба обґрунтувати положення мантис та порядків у заданих регістрах блока обробки данных, формат зображення чисел у ціх регістрах (кількість знакових та цифрових розрядів).

### 10.3. Методичні вказівки до виконання лабораторної роботи

При виконанні завдання варто прийняти, що ісходні дані (нормалізовані або рівні нулю числа) до початку операції знаходяться у внутрішніх регістрах регістрової пам'яті процесора у наступному порядку:

- R0 - модуль мантиси першого операнда A;
- R1 - порядок першого операнда A;
- R2 - знак першого операнда A (у старшому розряді);
- R3 - модуль мантиси другого операнда B;
- R4 - порядок другого операнда B;
- R5 - знак другого операнда B (у старшому розряді);

- R0 - модуль мантиси результату С;
- R1 - порядок результату С;
- R2 - знак результату С (у старшому розряді).

При реалізації алгоритму припустимо, що нульовий операнд у мікропроцесорі представляється нульовими кодами порядку і мантиси. При виникненні переповнення порядку або втраті значимості мантиси результату (Мс = 0) у полі порядку результату встановлюється код, який містить одиниці у всіх розрядах. При цьому у відповідності зі стандартом, при позитивному переповненні порядку результату у всіх розрядах мантиси результату встановлюються одиниці, а у знаковому розряді - "0" (плюс); при негативному переповненні - одиниці у всіх розрядах мантиси результату і "1" (мінус) - у знаковому розряді результату. Указівкою на втрату значимості мантиси результату виступає заперечність мантиси з нулем у всіх розрядах (при цьому у всіх розрядах порядку результату записуються одиниці).

При виборі формату даних і кількісті секцій мікропроцесора варто враховувати діапазон і точність зображення даних. Кількість цифрових розрядів порядку необхідно вибирати із співвідношення:

$$10^{k} < 2^{2^{m}}, \qquad (10.1)$$

де m - кількість цифрових розрядів порядку при підставі характеристики, рівної двом;
k - кількість цифрових розрядів порядку в десятковій системі.

Крім того, у форматі чисел необхідно передбачити розряд для збереження знака порядку. Кількість цифрових розрядів мантиси при підставі характеристики, рівної двом, вибирається із співвідношення:

$$10^{-\lambda} > 2^{-n}, \qquad (10.2)$$

де $\lambda$ - кількість цифрових розрядів мантиси у десяткової системі числення;
n - кількість цифрових розрядів у мантисі двійкового коду (при підставі характеристики, рівної двом).

При інших підставах характеристики попереднє співвідношення відповідним чином коректується. Отримане значення кількості розрядів остаточно округляється до чотирьох (ціла кількость секцій процесора).

У загальному випадку можливо виділити три основних етапи виконання операції складання - віднімання чисел у форматі з рухомою комою (див. рис. 10.1):

- порівняння та вирівнювання порядків;
- складання - віднімання мантис;
- нормалізація результату.

Розглянемо приклад виконання операції складання чисел з рухомою комою коли підстава характеристики S = 16 та порядок зображується з позитивним нулем (ПН). Припустимо, що формат даних має наступні поля: один байт використовується для збереження знака мантиси (один біт) і порядку (сім біт) і три байти (24 біта) - для запису розрядів мантиси.

Нехай початкові числа відповідно рівні: $A = (-24B5)_{16}$ і $B = (423)_{16}$. У нормалізованій формі ці числа мають вигляд:

$$\mathbf{A} = (-0,24B5)\,16^{+4}; \qquad \mathbf{B} = (0,423)\,16^{+3}.$$


Рисунок 10.1. Узагальнена блок-схема алгоритму складання-віднімання дійсних чисел

У двійковому коді порядки чисел відносно рівні: $P_A = +100$; $P_B = +011$.

Зміщені порядки чисел у форматі з позитивним нулем (зміщення E = 64) мають вигляд:

$$\mathbf{P_A^{64}} = 1\ 000\ 000 + 100 = 1\ 000\ 100;$$
$$\mathbf{P_B^{64}} = 1\ 000\ 000 + 011 = 1\ 000\ 011.$$

В цілому, числа у нормалізованій формі мають таке зображення:

**A:** 1. 100 0100 ‘ ‘ 0010 0100 1011 0101 0000 0000;
**B:** 0. 100 0011 ‘ ‘ 0100 0010 0011 0000 0000 0000.

Для порівняння порядків чисел треба віднімати від порядку числа А порядок числа В:

| | |
|---|---|
| $\underline{P_A^{64}} =$ | 1 . 000 100 |
| $+ \underline{P_B^{64}} =$ | 0 . 111 100 |
| $+1 =$ | 0 . 000 001 |
| $(P_A - P_B)_{ДК} = (1)$ | 0 . 000 001 |

"Знак" різниці порядків та перенос зі знакового розряду не збігаються. Отже, переповнення суматора порядків вітсутне і на виході суматора має місце коректна різниця порядків, яка дорівнює значенню плюс одиниці. Тому число В менше числа А і мантиса числа В повинна бути денормалізована шляхом зсуву на один розряд праворуч. Після зсуву мантиси числа В та збільшення його порядку на одиницю маємо:

**B:** 0 . 1 000 100 ' ' 0000 0100 0010 0011 0000 0000.

Після вирівнювання порядків, здійснюється операція складання мантис чисел А і В у додатковому коді:

| | |
|---|---|
| $(M_A)_{ДК} =$ | 1 . ' 1101 1011 0100 1011 0000 0000 |
| $+ (M_B)_{ДК} =$ | 0 . ' 0000 0100 0010 0011 0000 0000 |
| $(M_A + M_B)_{ДК} = (0)$ | 1 . ' 1101 1111 0110 1110 0000 0000. |

Перенос у знаковий розряд та вихідний перенос зі знакового розряду збігаються, тому переповнення суматора мантис вітсутне. Мантиса результату вийшла негативної, тому необхідно перейти до прямого коду:

$$(M_A + M_B)_{ПК} = 1 . ' 0010\ 0000\ 1001\ 0010\ 0000\ 0000.$$

Таким чином, результат є нормалізованим числом.

Кінцевий результат складання чисел А и В має вигляд:

$$A + B = 1 . ' 1\ 000\ 100\ ' '\ 0010\ 0000\ 1001\ 0010\ 0000\ 0000.$$

У 16 – ній системі результат дорівнює: $(A + B)_{16} = (-0,2092)\ 16^{+4}$.

Якщо сума мантиси після перетворення в прямий код є не нормалізованої, на останньому кроці здійснюється нормалізація результату. Нехай, наприклад, прямий код суми мантис і порядок результату мають вигляд:

$$A + B = 1 . 1\ 000\ 100\ ' '\ 0000\ 0101\ 0010\ 0000\ 0000\ 0000.$$

Нормалізація результату виконується шляхом зсуву мантиси ліворуч на один 16 - ий розряд і відняття одиниці від порядку. У результаті зсуву мантиси ліворуч одержимо:

$$(M_{A+B})_{ПК} = 1 . ' 0101\ 0010\ 0000\ 0000\ 0000\ 0000.$$

Зменшимо також порядок на одиницю. Для цього порядок підсумовуємо з доповненням зміщенної мінус одиниці:

$$
\begin{array}{rl|l}
 & P^{64}_{A+B} & = & 1.\ 000\ 100\ ' \\
+ & (-1)^{64} & = & 0.\ 111\ 111\ ' \\
\hline
 & (P_{A+B} - 1)_{ДК} & = (1) & 0.\ 000\ 011\ '.
\end{array}
$$

Результуючий порядок (+ 3) зображенний у додатковому коді. При цьому переповнення суматора порядків вітсутне тому що знаковий розряд суми та вихідний перенос зі знакового розряду суматора не збігаються. Для переходу до зміщеного порядку знаковий розряд отриманого додаткового коду необхідно інвертувати. Тоді остаточно порядок буде дорівнювати:

$$(P_{A+B})^{64} = 1.\ 000\ 011\ ' = (+3)^{64}.$$

Результат операції складання в нормалізованій формі у цьому випадку буде мати вигляд:

$$(A + B) = 1.\ 1\ 000\ 011\ '\ '\ 0101\ 0010\ 0000\ 0000\ 0000\ 0000$$

або

$$(A + B)_{16} = (-0,52)\ 16^{+3} = -520_{16}.$$

Процесорна секція К1804ВС2 має набір спеціальних функцій, які можуть бути використані при розробці мікропрограм операцій з рухомою комою [1 , 2]. Так, наприклад, функція нормалізації числа в додатковому коді (звичайної або подвійної довжини) виконується шляхом зсуву числа в сторону старших розрядів доти, поки два старших розряди не будуть мати різні значення. Якщо два старших розряди мають однакові значення, то мікрокоманда нормалізації не відбувається.

При нормалізації чисел звичайної довжини використовується регістр PQ процесорної секції. Число міститься в регістр PQ і з надходженням позитивного фронту сигналу CLK виконується зсув убік старших розрядів. Якщо усі розряди регістра PQ нульові, на виході Z процесорної секції ВС2 встановлюється сигнал логічної одиниці. У цьому випадку нормалізація не виробляється.

При розрахунку часових параметрів АП варто прийняти такі значення : $t_{BC1} = t_{BC2} = 85$ нс, $t_{BY4} = 125$ нс, $t_{BY1} = 102$ нс, $t_{IP1} = 21$ нс, $t_{KП12} = 25$ нс, $t_{MПП} = 70$ нс.

### 10.4. Зміст звіту

1. ГСМ та таблиця кодування роботи МОП (для БОД, СУАМ та інших блоків).
2. Приклад виконання операції над заданими числами.
3. Розрахунок часових параметрів пристрою.
4*. Листінг VHDL – проекту та результати експериментальних досліджень (факультативно).

### 10.5. Контрольні запитання

1. Яким чином виконується аналіз переповнення розрядної сітки машини, коли порядок результату зображений у вигляді:
   а) – додаткового кода;
   б) – кода з позитивним нулем;
   в) – кода з негативним нулем.
2. Яка буде реакція ЦОМ, коли мантиса результату прийме нульове значення ?

## Лабораторна робота № 11

# РОЗРОБКА ТА СИМУЛЯЦІЯ АЛГОРИТМІВ ОПЕРАЦІЙ МНОЖЕННЯ ТА ДІЛЕННЯ ЧИСЕЛ У ФОРМАТІ З РУХОМОЮ КОМОЮ

**МЕТА РОБОТИ:** вивчення алгоритмів множення і ділення чисел з рухомою комою та способів виконання симуляції цих операцій у секційних мікропроцесорах серії К1804.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

**В алгоритмі множення** дійсних нормалізованих чисел можуть бути виділені наступні основні етапи:

1. Перевірка мантис співмножників на рівність нулю. Якщо хоча б одна з мантис дорівнює нулю, формується код повного нуля (у всіх розрядах – нулі) і зводиться в одиничний стан відповідний прапор (FZ = 1).

2. Визначається порядок добутку шляхом підсумовування порядків співмножників у заданому коді (ДК, ПН, НН). У випадку виникнення переповнення формується сигнал переповнення і операція множення блокується (завершується).

3. Виробляється перемножування мантис чисел по одному з відомих алгоритмів множення з фіксованої комою.

4. Виконується нормалізація та округлення мантиси результату; у випадку виникнення переповнення розрядної сітки порядків формується сигнал переповнення (FV = 1).

**В алгоритмі ділення** варто дотримувати наступних етапів:

1. Якщо мантиса дільника дорівнює нулю, то формують код переповнення, зводять прапор переповнення (FV =1) та завершують операцію.

2. У випадку, коли мантиса діленого дорівнює нулю (а мантиса дільника відмінна від нуля), формується код нульового результату і зводиться відповідний прапор (FZ = 1).

3. Визначається порядок частки шляхом віднімання з порядку діленого порядка дільника.

4. Виконується ділення мантис по одному з відомих алгоритмів ділення з фіксованої комою дробових чисел.

5. Відбувається аналіз (і в разі потреби виконується) нормалізація результату і його округлення.

6. Виконується аналіз переповнення розрядної сітки суматора порядку і у випадку виникнення переповнення зводиться відповідний прапор.

### 11.1. Послідовність виконання лабораторної роботи

Розглянути всі етапи виконання операції множення (ділення) чисел у форматі з рухомою комою. Пункти послідовності виконання цієї роботи такі самі, як і у лабораторної роботі за № 10.

### 11.2. Варіанти індивідуальних завдань

Варіанти індивідуальних завдань наведені у табл. 11.1 – табл. 11.4. При реалізації ГСМ та таблиці кодування необхідно самостійно вибрати та обгрунтувати формати зображення операндів та результату виконання заданої операції.

Задана операція. Таблиця 11.1

| (N)mod2 | Тип операції |
|---|---|
| 0 | Множення |
| 1 | Ділення |

Тип процедури, яка розробляється індивідуально. Таблиця 11.2

| (N)mod3 | Тип процедури операцій множення або ділення |
|---|---|
| 0 | Обчислювання порядку результату |
| 1 | Обчислювання мантиси результату |
| 2 | Виконання нормалізації результату |

Алгоритм множення мантис (у прямому коді). Таблиця 11.3

| (N)mod4 | Назва алгоритму множення |
|---|---|
| 0 | "А" |
| 1 | "Б" |
| 2 | "В" |
| 3 | "Г" |

Алгоритм ділення мантис (у прямому коді). Таблиця 11.4

| (N)mod2 | Назва алгоритму ділення |
|---|---|
| 0 | "а" |
| 1 | "б" |

### 11.3 Методичні вказівки до виконання лабораторної роботи

При виконанні завдання будемо припускати, що ісходні числа (нормалізовані або рівні нулю) до моменту початку операції розташовані у внутрішніх регістрах процесорного блоку ВС1 або ВС2:

R0 - модуль мантиси першого числа А;
R1 - порядок першого числа А;
R2 - знак першого числа А;
R3 - модуль мантиси другого числа В;
R4 - порядок другого числа В;
R5 - знак другого числа В.

Результат виконання операції після її завершення повинний бути розташований у регістрах загального призначення на місці числа А.

Блок - схема операції повинна включати наступні основні етапи: перевірку операндів на нуль і формування відповідного результату при нульовому операнді

(операндах) без виконання операції; підсумовування (при операції множення) або віднімання (при операції ділення) порядків з контролем негативного або позитивного переповнення сумматора порядків і формуванням відповідного результату без виконання операції; множення (ділення) мантис по одному з відомих алгоритмів; контроль порушення і відновлення нормалізації отриманого результату; аналіз переповнення сумматора порядків при коректуванні порядку в процесі нормалізації мантиси результату. Граф – схему алгоритму виконання заданої операції розробити самостійно.

При реалізації алгоритму для спрощення мікропрограм доцільно використовуватися регістром - акумулятором мікропроцесора (PQ) для розміщення множника (частки). Для організації лічильника циклів при множенні (діленні) можливо використовувати один з регістрів мікропроцесорів ВС1 (або ВС2). Запис константи в лічильник циклів може бути здійснений з відповідного поля данных регістра мікрокоманд.

У кожному циклі множення (або ділення) вміст лічильника варто зменшувати на одиницю. Після того, коли він прийме нульове значення на черговому такті роботи блоку мікропрограмного керування, варто зробити вихід з циклу. В іншому варіанті лічильник тактів може бути організований з використанням внутрішніх регістрів – лічильників мікросхем ВУ4 (або ВУ1). У цьому випадку підрахунок кількості циклів буде вироблятися паралельно з обчисленням суми часткових добутків (залишку), що дозволить скоротити час виконання всієї операції.

У таблиці кодування мікропрограмної пам'яті в полі коментарів варто передбачити два розділи: поле для опису мікрооперації в блоці обробки даних і поле для опису мікрооперації в блоці мікропрограмного керування в поточному такті роботи.

Розглянемо приклад виконання операції множення з рухомою комою при підставі характеристиці S = 2 та представленні порядку з негативним нулем (НН). Нехай при цьому формат даних має наступні поля: один байт використовується для збереження знака числа (1-й біт - старший) і порядку (сім біт) та три байти (24 біта) - для зображення модуля мантиси.

Нехай співмножники відповідно рівні: $A = (-16)_{16}$; $B = (+1A)_{16}$.

У нормалізованій формі ці числа мають вигляд:

$$\mathbf{A} = -0,10110 \cdot 2^{+5}; \qquad \mathbf{B} = +0,11010 \cdot 2^{+5}.$$

У двійковому коді порядки відносно рівні: $P_A = P_B = +101$.

Зміщені порядки з негативним нулем (зміщення E = 63) мають вигляд:

$$\mathbf{P^{63}_A = P^{63}_B} = 1\ 000\ 100.$$

У цілому, числа в обраному форматі будуть мати наступний вигляд:

**A** : 1 . 1 000 100 ‘ ‘ 1011 0000 0000 0000 0000 0000 ;
**B** : 0 . 1 000 100 ‘ ‘ 1101 0000 0000 0000 0000 0000 .

Для обчислення порядку добутку визначимо суму зміщених порядків:

| | |
|---|---|
| $\mathbf{P^{63}_A} =$ | $1\ 000\ 100 = (+5)^{63}$ |
| $+\ \mathbf{P^{63}_B} =$ | $1\ 000\ 100 = (+5)^{63}$ |
| $+\mathbf{1} =$ | $0\ 000\ 001$ |
| $(\mathbf{P_A + P_B - 1})_{дк} = (1)$ | $0\ 001\ 001 = (+9)_{дк}$ |

Знаковий розряд суми порядків з негативним нулем не збігається з вихідним переносом, тому переповнення суматора порядків вітсутне. Отже, на виході суматора має місце коректне значення суми порядків (сума порядків мінус одиниця у додатковому коді). Для одержання значення зміщенного порядку добутку необхідно виконати інвертування старшого розряду суми порядків:

$$(P_A + P_B)^{63} = 1\ 001\ 001\ ' = (+10)^{63}.$$

Множення модулів мантис можливо виконувати по одному з відомих алгоритмів, наприклад, по алгоритму "А" (тут приводиться лише кінцевий результат):

$$\begin{array}{r} |M_A| = '\ 1011\ 0000\ 0000\ 0000\ 0000\ 0000 \\ |M_B| = '\ 1101\ 0000\ 0000\ 0000\ 0000\ 0000 \\ \hline |M_A \times M_B| = '\ 1000\ 1111\ 0000\ 0000\ 0000\ 0000 \end{array}$$

Знак результату ( ЗнС ) визначається шляхом складання по модулю двох знакових розрядів чисел А і В:

$$\mathbf{ЗнС = ЗнА \oplus ЗнВ} = 1 \oplus 0 = 1.$$

Таким чином, остаточно нормалізований результат має вигляд:

$$C = 1.1\ 001\ 001\ '\ '\ 1000\ 1111\ 0000\ 0000\ 0000\ 0000.$$

При множенні нормалізованих мантис може виникнути порушення нормалізації праворуч на один розряд:

$$|M_A| \times |M_B| = '\ 01XX\ XXXX\ XXXX\ XXXX\ XXXX\ XXXX.$$

У цьому випадку на останньому етапі операції виконується нормалізація результату шляхом зсуву мантиси добутку ліворуч та зменшення порядку на одиницю. У приведеному вище прикладі після зсуву одержимо:

$$|M_A| \times |M_B| = '\ 1XXX\ XXXX\ XXXX\ XXXX\ XXXX\ XXXX.$$

Якщо при цьому порядок результату буде, наприклад, дорівнювати

$$(P_A + P_B)^{63} = 1.\ 001\ 001\ ' = (+10)^{63},$$

то його корекція зведиться до наступної дії:

$$\begin{array}{rrl} & (P_A + P_B)^{63} = & 1.\ 001\ 001\ ' = (+10)^{63} \\ + & (-1)^{63} = & 0.\ 111\ 110\ ' = (-1)^{63} \\ & +1 = & 0.\ 000\ 001\ ' = (+1) \\ \hline & (P_A + P_B - 1)_{дк} = (1) & 0.\ 001\ 000\ ' = (+8)_{дк} \end{array}$$

Після інвертування старшого розряду одержимо наступний результат:

$$(P_A + P_B - 1)^{63} = 1.\ 001\ 000\ ' = (+9)^{63}.$$

Операція ділення починається з віднімання порядків. Нехай ділене (A) та дільник (B) відповідно рівні:

$$\mathbf{A}: \quad 1.1\ 001\ 001\ `\ `\ 1000\ 1111\ 0000\ 0000\ 0000\ 0000;$$
$$\mathbf{B}: \quad 0.1\ 001\ 001\ `\ `\ 1101\ 0000\ 0000\ 0000\ 0000\ 0000.$$

У цьому випадку різниця порядків може бути обчислена у такому способі:

$$\begin{array}{rl} \underline{P_A}^{63} = & 1\ 001\ 001\ ` \\ P_B^{63} = & 0\ 110\ 110\ ` \\ \hline (P_A - P_B - 1)_{ДК} = (0) & 1\ 111\ 111\ `. \end{array}$$

Зміщений порядок результату утвориться шляхом інвертування знакового розряду отриманого додаткового коду:

$$(P_A - P_B)^{63} = 0\ 111\ 111 = (+0)^{63}.$$

Розділивши мантиси (по одному з відомих алгоритмів) і обчисливши знак частки, остаточно одержимо частку (в заданому форматі):

$$\mathbf{C} = 1.0\ 111\ 111\ `\ `\ 1011\ 0000\ 0000\ 0000\ 0000\ 0000.$$

При діленні модулів мантис може виникнути порушення нормалізації як ліворуч, так і праворуч. Відновлення нормалізації здійснюється зсувом мантиси частки відносно праворуч або ліворуч та корекції порядку частки відповідно на "-1" або "+1". Остаточне значення порядку формується (при вітсутності переповнення) шляхом інвертування знакового розряду отриманого коду.

### 11.4. Зміст звіту

Пункти звіту цієї лабораторної роботи такі самі, як і у лабораторної роботі за № 10.

### 11.5. Контрольні запитання

1.Яким чином виконується округлення отриманого результату операції множення або ділення дійсних чисел ?

2. Яким чином у БОД на основі секцій ВС1 сформувати ознаку переповнення суматору порядків у коді:

а) з негативним нулем ? ;

б) з позитивним нулем ?

3. Як виявити порушення нормалізації результату у додатковому коді ?

## Лабораторна робота № 12

# АЛГОРИТМИ ОБРОБКИ КОМАНД ДОВІЛЬНОГО ФОРМАТУ В ЦЕНТРАЛЬНОМУ ПРОЦЕСОРНОМУ ПРИЛАДІ (ЦПП) ТА СИМУЛЯЦІЯ РОБОТИ ЦПП

**МЕТА РОБОТИ**: вивчення форматів команд різного типу та придбання навичок мікропрограмування та симуляції алгоритмів їх обробки.

## ЗАГАЛЬНІ ПОЛОЖЕННЯ

У найбільш загальному випадку в складі мікроЕОМ можуть бути виділені наступні основні блоки: арифметично – логічний пристрій (АЛП), основна оперативна пам'ять (ОП), центральний пристрій керування (ЦПК), пульт керування (ПУ).

АЛП реалізує заданий набір операцій (складання, множення, ділення та інші) і може бути виконаний на основі процесорних секцій ВС1 або ВС2. В оперативній пам'яті машини зберігаються машинні команди програми, операнди та деякі проміжні дані. ЦПК формує послідовність керуючих сигналів, під впливом яких відбувається вибірка поточної команди програми з ОП, декодування її полів, вибірка операндів та виконання заданої операції.

ЦПК може бути реалізований як автомат з жорсткою логікою (за схемою Милі або Мура), або як автомат з програмувальною логікою. В останньому випадку ЦПК відносять до автоматів з мікропрограмним управлінням, до складу яких звичайно входять мікросхеми керування послідовністю мікрокоманд (наприклад, ВУ4 або ВУ1), а також мікропрограмна пам'ять (МПП). МПП, як правило, реалізується у вигляді мікросхем постійної пам'яті (ROM), у яких зберігаються мікропрограми (сукупність мікрокоманд). Сукупність мікрокоманд ROM керують виконанням команд в ЕОМ.

АЛП та ОП можуть мати власні місцеві пристрої керування, які взаємодіють з ЦПК за допомогою інформаційних сигналів. При іншому варіанті ЦПК може виконувати також функції місцевих пристроїв керування АЛП і ОП. Для цієї мети у форматі мікрокоманди ЦПК повинні бути відповідні поля для керування зазначеними блоками.

### 12.1. Послідовність виконання лабораторної роботи

1.Вивчити з використанням методичних вказівок склад мікроЕОМ та алгоритми виконання типових команд одиничного та подвійного форматів.

2. Розробити функціональну схему, ГСМ та таблицю кодування вибірки з ОП та виконання заданої команди.

3. Розрахувати час вибірки команд з ОП та виконання поточної команди. Час циклу запису (читання) оперативної пам'яті toп прийняти рівним 500 нс.

4*. Розробити VHDL – модель мікроЕОМ на основі мікросхем набора К1804. Виконати налагодження VHDL - проекту та провести його симуляцію (цей пункт роботи виконується факультативно).

5. Скласти звіт про виконану лабораторну роботу та захистити його.

6. Дати відповіді на контрольні запитання.

### 12.2. Варіанти індивідуальних завдань

Формат заданої команди наведений у табл. 12.1, код зображення операндів – у табл.12.2, операція АП - у табл. 12.3. Формати команд мікроЕОМ визначені на рис.12.1. Лічильник адреси команд (ЛЧАК) необхідно вибирати виходячи з

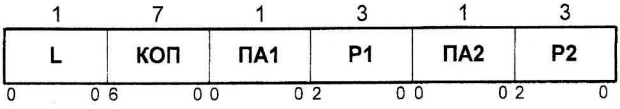
табл.12.4. Організація модуля оперативної пам'яті (ОП) наведена у табл. 12.5. Блок мікропрограмного керування реалізувати на мікросхемах, які наведені у табл. 12.6.

Визначення довжини команди. Таблиця 12.1

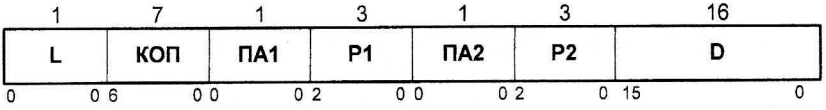
| (N)mod2 | Формат команди |
|---|---|
| 0 | L = 0 ("коротка" команда) |
| 1 | L = 1 ("довга" команда) |

Код для зображення операндів. Таблиця 12.2

| (N)mod2 | Код операнда |
|---|---|
| 0 | ПК (прямий код) |
| 1 | ДК (додатковий код) |




а)




б)

Рисунок 12.1. Формати команд:
а) — "коротка" команда (L = 0)
б) — "довга" команда (L = 1)

Арифметична операція поточної команди та її код. Таблиця 12.3

| (N)mod8 | Операція | КОП | АП |
|---|---|---|---|
| 0 | a x b (алг. А) | 027 | 112 |
| 1 | a x b (алг. Б) | 073 | 57 |
| 2 | a x b (алг. В) | 103 | 76 |
| 3 | a x b (алг. Г) | 96 | 51 |
| 4 | $a^n / b^n$ (алг. а) | 105 | 27 |
| 5 | $a^n / b^n$ (алг. б) | 029 | 115 |
| 6 | $a^{2n+1} / b^n$ (алг. а) | 127 | 116 |
| 7 | $a^{2n+1} / b^n$ (алг. б) | 97 | 121 |

**Примітки:** КОП – код операції регістра команд (РК);
АП – початкова адреса процедури обробки команди;
a , b - операнди (у прямому або додатковому коді).

Розташування лічильника адреси команд. Таблиця 12.4

| (N)mod2 | Розташування ЛЧАК |
|---|---|
| 0 | У зовнішньому регістрі |
| 1 | У одному з регістрів РЗП |

Ємність модуля оперативної пам'яті (ОП). Таблиця 12.5

| (N)mod4 | Організація ОП |
|---|---|
| 0 | 4 K x 8 |
| 1 | 8 K x 16 |
| 2 | 16 K x 32 |
| 3 | 8 K x 8 |

Тип мікросхеми для реалізації блока мікропрограмного керування. Таблиця 12.6

| (N)mod2 | Мікросхема для реалізації БМК |
|---|---|
| 0 | К1804ВУ1 |
| 1 | К1804ВУ4 |

Адресації команд. Таблиця 12.7

| L | ПА1 або ПА2 | Тип адресації |
|---|---|---|
| 0 | 0 | Пряма адресація у "короткій" команді<br>ОР = (Р1) |
| 0 | 1 | Непряма адресація у "короткій" команді<br>ОР = ОП [ (Р1) ] |
| 1 | 0 | Пряма адресація у "довгій" команді<br>ОР = (Р1) + D |
| 1 | 1 | Непряма адресація у "довгій" команді<br>ОР = ОП [ (Р1) + D ] |

**Примітки:** ОР - операнд; Р1 або Р2 - [000, 001, 010, 011, 100, 101, 110, 111] – один з перших восьми РЗП, які використовуються мікроЕОМ (один із них може бити ЛЧАК, наприклад, R7). РЗП від R8 до R15 використовуються як регістри арифметичного пристрою (наприклад, R8 та R9 для операндів, а R10 - для результату операції).

### 12.3. Методичні вказівки до виконання лабораторної роботи

Розглянемо виконання операції "Складання" у додатковому коді на прикладі гіпотетичної команди формату "регістр – пам'ять" (L = 0, ПА1 = 0, ПА2 = 0). В команді перші 10 розрядів - поле коду операції (КОП) регістра команд (РК). Поле задає значення параметрів L, ПА1, ПА2 та машинну операцію, яка повинна бути реалізована. Наступне трьох-розрядне поле Р1 вказує адресу першого операнда (адресу регістра загального призначення АЛП), поле Р2 задає адресу другого операнда, яка знаходиться в РЗП (сам операнд розташований в оперативній пам'яті (ОП) машини). Результат операції підсумовування двох операндів записується в РЗП, адреса якого вказана у полі Р1 (записується на місце першого операнда).

Узагальнена структурна схема мікроЕОМ для реалізації заданої команди наведена на рис. 12.2. Ємність оперативної пам'яті складає 64К х 16 розрядів. Це означає, що в однієї комірці ОП розташовується одна команда формату "регістр - пам'ять". Приймемо, що операнди мають також 16 розрядів і займають одну комірку пам'яті.

До складу мікроЕОМ входять наступні основні блоки:

ЛЧАК - лічильник адреси команд;
РК - регістр команд;
ОП - оперативна пам'ять;
БОД - блок обробки даних;
БМК - блок мікропрограмного керування;
ПК - пульт керування та індикації;
БС - блок синхронізації.

Зв'язок окремих блоків мікроЕОМ здійснюється через системний інтерфейс, утворений трьома магістралями:

U - керування; A - адреси; D - даних.

Для керування мікроЕОМ використовується пульт керування (ПК), через який оператор може виконувати наступні операції:

- заносити в ЛЧАК початкову адресу програми $A_{поч}$;
- здійснювати запуск процесора шляхом натискання кнопки "Пуск";
- виконувати зупинку роботи процесора за допомогою тумблера "Зупинка".

Після запуску ЕОМ, блок синхронізації (БС) починає видавати тактові сигнали CLK, під впливом яких БМК формує послідовність керуючих сигналів $Y_i$. Останні керують системною магістраллю та усіма блоками машини. Структура БМК наведена на рис. 12.3.

Код, обраної з ОП команди (КОП), за допомогою інструкції JMAP ("вхід у команду") дешифрується ФПА (формувач початкової адреси) та у вигляді початкової адреси мікропрограми АП надходить у схему керування послідовністю мікрокоманд (СКАМ) з використанням мікросхеми ВУ4. Сформована на виході Y СКАМ адреса надходить у МПП. Обрана з МПП мікрокоманда по фронту тактового сигналу CLK записується в регістр мікрокоманд (РМК). Виходи РМК використовуються у вигляді керуючіх сигналів, які надходять в всі блоки машини.

Оперативна пам'ять (ОП) необхідна для збереження програм і даних. У її складі можна виділити наступні елементи:

PDO - регістр вихідних даних;
PDI - регістр вхідних даних;
РАП - регістр адреси пам'яті;
МПКП - місцевий пристрій керування пам'яттю.

Робота ОП у режимах запису та читання інформації (табл. 12.8) здійснюється шляхом формування в МПКП (з шини U) керуючих сигналів:

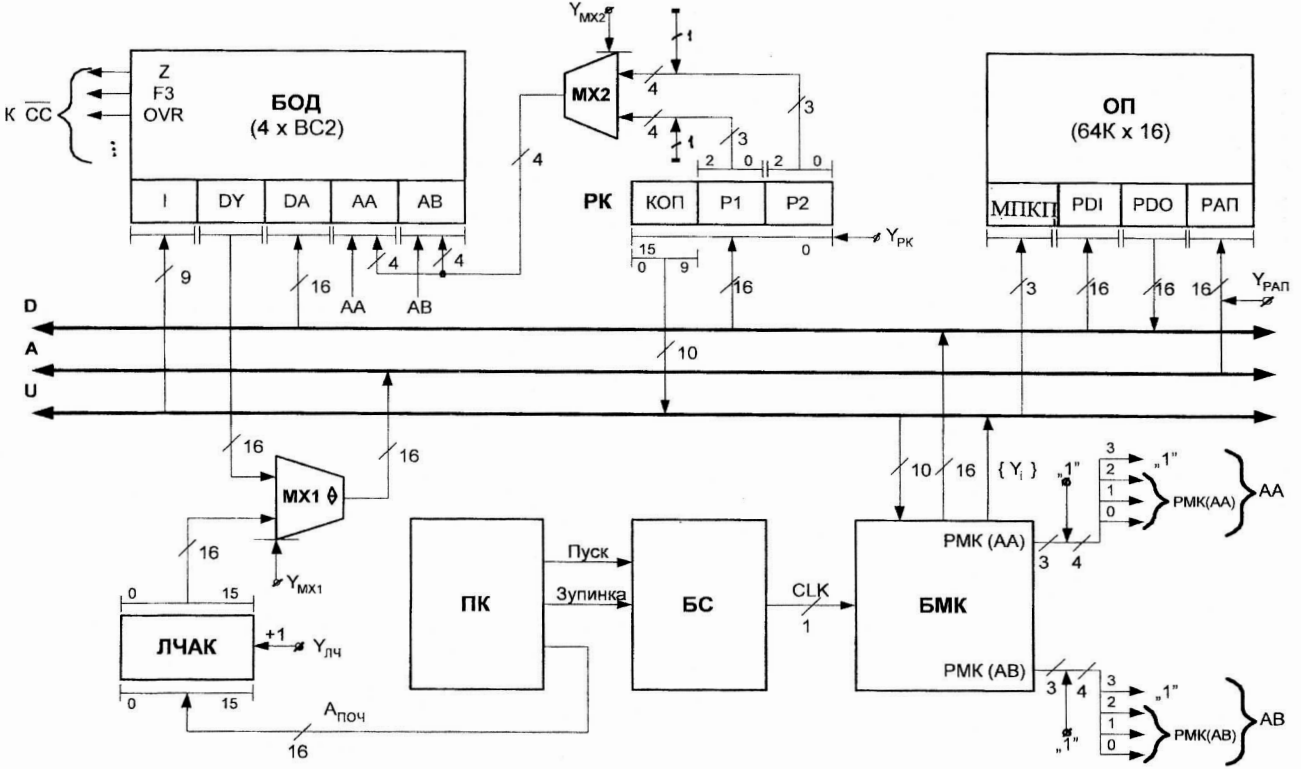



Рисунок 12.2. Узагальнена структурна схема мікроЕОМ

$\overline{MEMR}$ - сигнал дозволу читання з ОП;
$\overline{MEMW}$ - сигнал дозволу запису в ОП;
ST - сигнал запуску МПКП (початок роботи ОП).

Режими роботи оперативної пам'яті машини. Таблиця 12.8

| ST | $\overline{MEMR}$ | $\overline{MEMW}$ | Режими роботи ОП |
|---|---|---|---|
| 0 | 0 | 1 | Читання |
| 0 | 1 | 0 | Запис |
| 1 | * | * | Збереження (PDO у третьому стані) |

Блок обробки даних (БОД) може бути реалізований з декількох процесорних секцій К1804ВС1 або К1804ВС2. Для послідовного доступу до РЗП з використанням адрес портів АА або АВ (у полях Р1 та Р2) використовується мультиплексор МХ2. Алгоритм робота МХ2 наведено в табл. 12.9.

Алгоритм роботи мультиплексора МХ2. Таблиця 12.9

| Умх2 | Вихід МХ2 |
|---|---|
| 0 | Р2 (РК) |
| 1 | Р1 (РК) |

Таблиця кодування вибірки команди "складання" формату "регістр - пам'ять" і виконання цієї команди наведені в табл.12.10. У таблиці наведено фрагмент мікропрограми після завершення вибірки команди в РК і розшифровки полів L, ПА1, ПА2. Таблиця складається з п'яти мікрокоманд. У мікрокоманді з адресою 10 здійснюється завантаження ЛЧАК у РАП по керуючому сигналу Урап. При цьому в БОД виконується операція NOP ("порожня операції"), а в БМК - інструкція CONT ("продовження").

При виконанні мікрокоманди з адресою 11 із ОП за адресою розташованою в РАП вибирається в регістр PDO команда, яка потім по сигналі Урк пересилається в регістр команд РК. У блоках БМК і БОД при цьому виконуються відповідно інструкції NOP і CONT.

У мікрокоманді с адресою 12 у БМК за допомогою інструкції JMAP (0010) відбувається вибірка з МПП мікрокоманди з адресою 31. Одночасно в цьому ж такті по сигналу Улч відбувається збільшення вмісту ЛЧАК на одиницю, тобто підготовляється адреса наступної команди програми, розташованої в ОП.

У мікрокоманді з адресою 31 поле РК (Р2) пересилається на вхід АА ВС2 і з РЗП, номер якого зазначений у полі Р2, витягається на шину DY БОД (ВС2) адреса другого операнда, яка через мультиплексор МХ1 (табл. 12.11) пересилається в РАП.

Алгоритм роботи мультиплексора МХ1. Таблиця 12.11

| Умх1 | Вихід МХ1 |
|---|---|
| 0 | DY |
| 1 | ЛЧАК |

У мікрокоманді з адресою 32 відбувається вибірка другого операнда з ОП у БОД за схемою: PDO → D → DA. Одночасно з цим поле РК (Р1) → АВ і перший

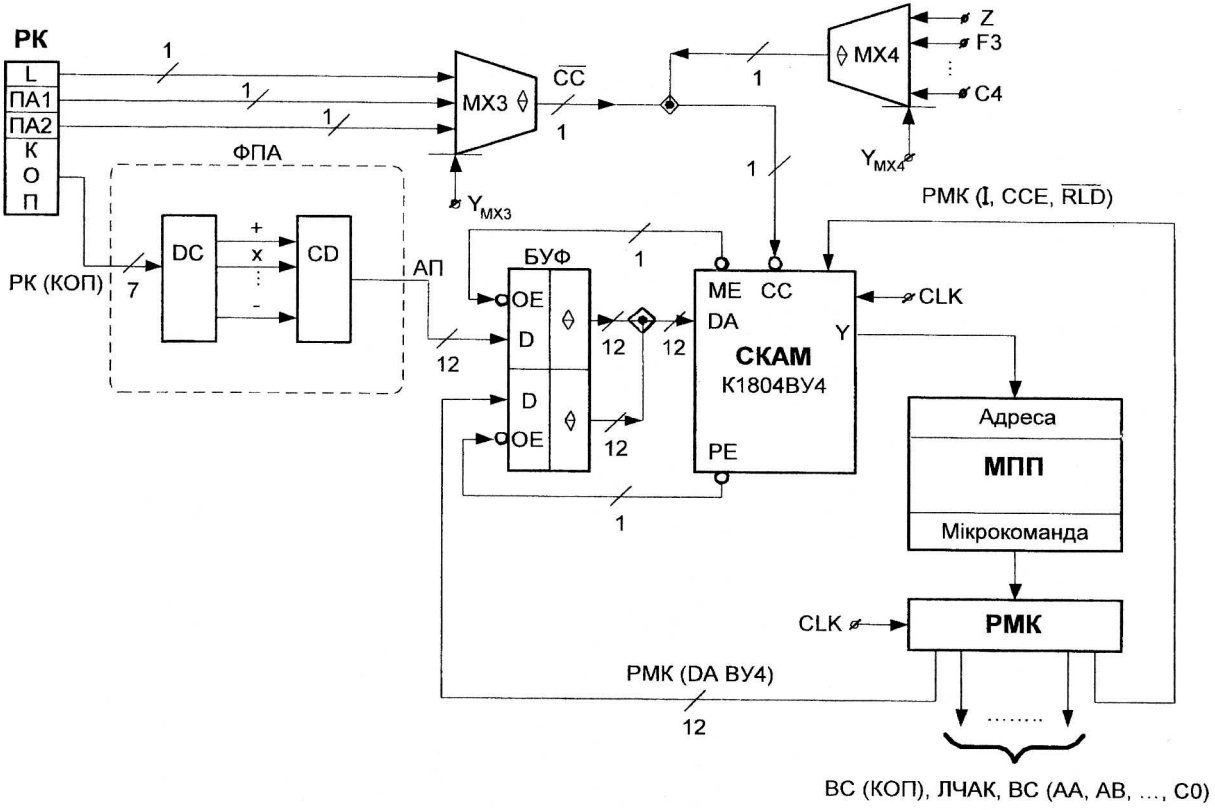



Рисунок 12.3. Структура БМК АП та ЦПК ЕОМ

Таблиця кодування мікропрограми виконання команди “складання”. Таблиця 12.10

| Адреса мікрокоманди | МІКРОКОМАНДА | | | | | | | | | | | | | | | | | | | | | Примітки |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Керування К1804ВУ4 | | | | Керування К1804ВС2 | | | | | | | | | | РАП | ОП | | РК | МХ1 | ОП | МХ2 | ЛЧАК | |
| | $\overline{OE}$ | $\overline{RLD}$ | I | DA | I 8765 | I 4321 | I 0 | $\overline{EA}$ | $\overline{OEB}$ | $\overline{OEY}$ | C0 | AA | AB | $\overline{IEN}$ | Урап | $\overline{ST}$ | $\overline{MEMR}$ | $\overline{MEMW}$ | Урк | Умх1 | Ум | Умх2 | Улч | |
| 10 | 0 | 1 | E | * | 1100 | **** | 0 | 0 | 0 | 0 | 0 | * | * | 1 | 0 | 1 | * | * | 1 | 1 1 | 1 | * | 1 | ЛЧАК → РАП |
| 11 | 0 | 1 | E | * | 1100 | **** | 0 | 0 | 0 | 0 | 1 | * | * | 0 | 1 | 0 | 0 | 1 | 0 | 1 1 | 0 | * | 1 | Чт.ОП, D → РК |
| 12 | 0 | 1 | 2 | АП | 1100 | **** | 0 | 0 | 0 | 0 | 0 | * | * | 1 | 1 | 1 | * | * | 1 | 1 1 | 1 | * | 0 | Декодування АП<br>ЛЧАК+1→ ЛЧАК |
| 31 | 0 | 1 | E | * | 1100 | 0110 | 0 | 0 | 0 | 0 | 0 | (P2) | * | 1 | 0 | 1 | * | * | 1 | 0 1 | 1 | 0 | 1 | РОН(P2) → РАП |
| 32 | 0 | 1 | E | * | 0100 | 0011 | 0 | 1 | 0 | 0 | 0 | * | (P1) | 0 | 1 | 0 | 0 | 1 | 1 | 1 1 | 0 | 1 | 1 | Чт. ОП,<br>D + (P1) → (P1) |

операнд вибирається з РЗП за адресою, зазначенної в полі Р1 команди. В АЛП над операндами виконується операція підсумовування. Результат записується в РЗП на місце першого операнда за адресою РК (Р1).

Керуючий сигнал Ym дозволяє збільшити тривалість сигналу CLK при роботі з "повільної" ОП [1, 2].

### 12.4. Зміст звіту

1. Функціональна схема мікроЕОМ з необхідними поясненнями окремих блоків та сигналів схеми.
2. ГСМ вибірки з ОП та виконання команди.
3. Таблиця кодування з коментарями.
4*. Результати симуляції роботи мікроЕОМ (факультативно).

### 12.5. Контрольні запитання

1. Яким чином та в якій послідовності відбувається обробка ознаки типу адреси:
   а) - непрямої; б) – прямої; в) – безпосередньої.
2. Що таке індексація ? Яким чином у ЦОМ відбувається обробка масивів данних ?
3. Формат команди має наступні поля: кода операції (КОП), ознаки адреси (ПА), ознаки індексації (ПІ), ознаки запису результату в ОП (ПЗ) та адреси операнда (А). У якій послідовності треба розглядати та виконувати поля такої команди після того, як вона записана з ОП в регістр команд ?

# ДОДАТОК

## Д1. МЕТОДИЧНІ ВКАЗІВКИ ДО ПРОЕКТУВАННЯ ТА МОДЕЛЮВАННЯ БЛОКІВ ОБРОБКИ ДАНИХ

### Д1.1 Мікропроцесорна секція К1804ВС1

Мікропроцесорна секція (МПС) К1804ВС1 призначена для приймання, оперативного збереження і обробки двійкової інформації. Одна секція має чотири розряди. При з'єднанні **n** МПС можлива обробка **4n** – розрядних слів з послідовним або паралельно – паралельним переносом ( при застосуванні схеми прискореного переносу (СПП).

Функціональна схема однієї процесорної секції наведена на рис.Д1.1. У її склад входять: арифметико-логічний пристрій (АЛП), регістровий запам'ятовуючий пристрій (РЗП), два зсувача ( ЗСВF і ЗСВQ ), два буферних регістри ( РА і РВ ), чотири мультиплексора (MXR, MXS, MXQ і MXY ), шинний формувач з трьома станами (БПУ), регістр-акумулятор ( PQ ) та блок управління ( БУ ).

На функціональній схемі використовуються наступні позначки сигналів:

**AA(3-0)** – входи адреси РЗП по каналу А;

**AB(3-0)** – входи адреси РЗП по каналу В;

**I(8 – 0)** - входи коду інструкції МПС;

**CLK** – тактовий сигнал синхронізації МПС;

**PF3, PF0** – двонаправлені ланки зсуву інформації зсувача ЗСВF;

**PQ3, PQ0** – двонаправлені ланки зсуву інформації зсувача ЗСВQ;

**D(3-0)** – інформаційний вхід;

**Y(3-0)** – інформаційний вихід з трьома станами;

**C0** – вхід переноса АЛП;

**C4** – вихід переноса АЛП;

**Z** – вихід ознаки нульового стану АЛП (з відкритим колектором );

**OVR** – вихід ознаки переповнення АЛП;

**F3** – вихід третього (старшого) розряду АЛП (знака результата). Якщо в АЛП використовується декілька МПС, то знаком результату буде на клемі F3 самої старшої МПС;

**P, G** - виходи умови поширення переносу і генерації місцевого переносу;

**$\overline{OE}$** - вхід для керування виходом **Y** буферного підсилювача з трьома станами (БПУ).

Розглянемо більш докладно елементи функціональної схеми ВС1.

***Регістровий ЗП*** містить 16 регістрів загального призначення (R0, R1, ... , R15) по чотири розряду в кожному (3 – 0). Повна функціональна схема РЗП наведена на рис.Д1.2. Стани адресних портів АА і АВ подаються на відповідні дешифратори DCA і DCB. Виходи цих дешифраторів керують роботою схем запису і читання інформації. Будь-які два слова РЗП, обумовлені станами адресних портів, можуть бути видані на відповідні порти А і В. Якщо на адресні порти подати однакові адреси, то на виходах А і В з'являться одинакові дані.

Запис результату з виходу шини **FC** (виходу зсувача ЗСВF) може відбуватися тільки через завдання адреси регістра по каналу АВ протягом низького рівня сигналу синхронізації ($\overline{CLK}$). Регістри РЗП виконані на DE – тригерах типу "заскочка". Инверсне значення сигналу ($\overline{CLK}$) подається на вхід **E** тригера. Для надійного запису інформації в РЗП (момент часу 23 на діаграмі рис. Д1.4.) адреса порту АВ і вміст шини FC повинні випереджати початок сигналу **CLK** і протягом дії низького рівня тактового сигналу не повинні змінюватися.

***Буферні регістри РА і РВ*** виконані на DE -тригерах. Однак, на відміну від тригерів РЗП, запис інформації в порти А і В відбувається за верхнього рівня тактового сигналу

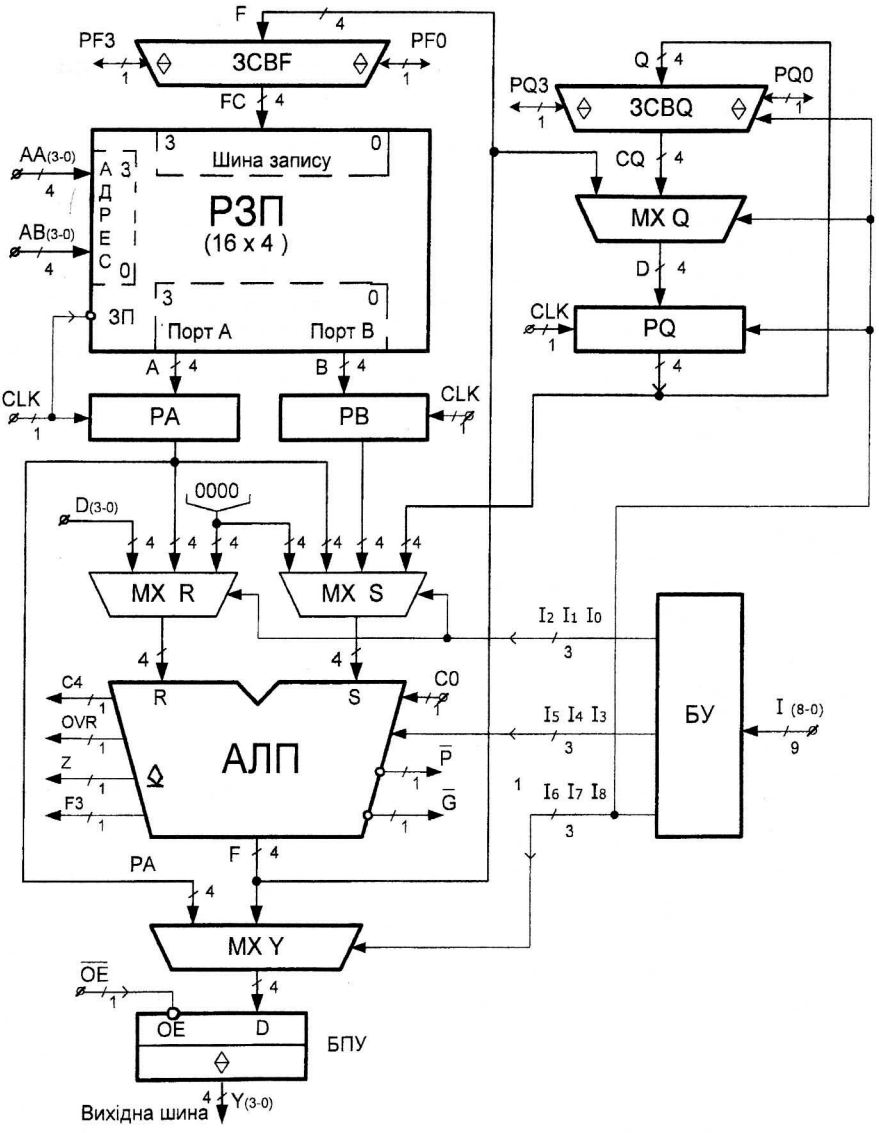



Рисунок Д1.1 Функціональна схема процесорного елемента К1804ВС1

**CLK** (момент часу 12 на діаграмі). Для коректної роботи схеми адреси AA і AB повинні бути стабільними протягом усього такту (інтервал часу 13). Якщо сигнал **CLK = 1** (момент

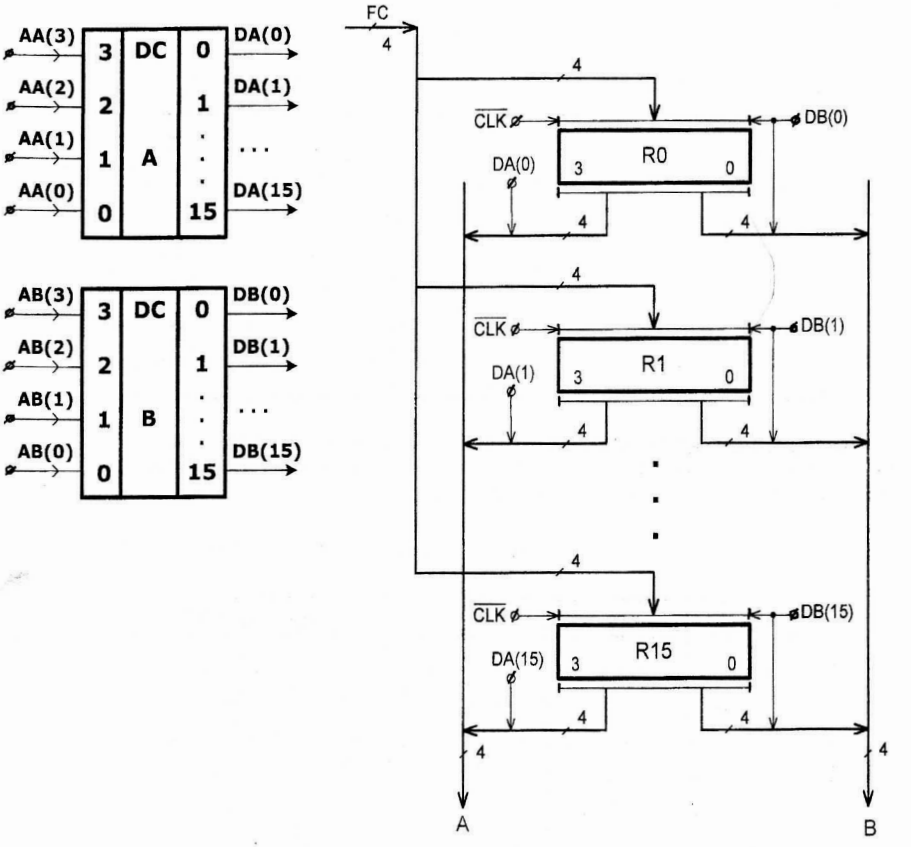



Рисунок Д1.2. Функціональна схема РЗП

часу 12), регістри РА і РВ приймають значення портів А і В (режим запису), а протягом інтервалу часу 23 ці регістри переходять у режим збереження інформації. Отже, на регістрах РА і РВ інформація залишається незмінною протягом інтервалу часу 23, хоча входи портів А и В можуть змінити свої стани в момент часу 23.

***Зсувач ЗСВF***. Інформація з виходу АЛП (шина F) перед записом у РЗП при передачі через ЗСВF може бути зсунута на один розряд ліворуч, праворуч, або передана без зсуву. Робота зсувача наведена на рис. Д1.3. Двонаправлені лінії з трьома станами **PF3** і **PF0** дозволяють здійснити видачу спадаючого при зсуву розряду або занести біт у розряд, який звільнюється при зсуву. При відсутності операції зсуву ці клеми вимкнуті і знаходяться в третьому стані.

***Комутатори входів R і S*** призначені для визначення джерел операндів, які беруть участь в операції АЛП. Комутатори реалізовані у вигляді двох мультиплексорів (MXR і MXS, див. рис.Д1.1). Як джерела операндів можуть бути обрані буферні регістри РА і (або)

РВ, регістр-акумулятор РQ, код зовнішньої шини D і код нуля (0000). Вибір джерел операндів задається розрядами **I2 – I0** інструкції МПС (див. табл.Д1.1).

Визначення джерел операндів АЛП. Таблиця Д1.1

| $I_2$ | $I_1$ | $I_0$ | R | S |
|---|---|---|---|---|
| 0 | 0 | 0 | PA | PQ |
| 0 | 0 | 1 | PA | PB |
| 0 | 1 | 0 | 0 | PQ |
| 0 | 1 | 1 | 0 | PB |
| 1 | 0 | 0 | 0 | PA |
| 1 | 0 | 1 | D | PA |
| 1 | 1 | 0 | D | PQ |
| 1 | 1 | 1 | D | 0 |

***Арифметико-логічний пристрій*** виконує три арифметичних і п'ять операцій бульовою логіки над операндами **R** і **S**, що надходять з виходів комутаторів. Роботою блоку АЛП керують розряди **I5 – I3** кода інструкції МПС (див. табл. Д2).

Визначення операції АЛП. Таблиця Д1.2

| $I_5$ | $I_4$ | $I_3$ | Операція АЛП |
|---|---|---|---|
| 0 | 0 | 0 | $F = R + S + C0$ |
| 0 | 0 | 1 | $F = S + \overline{R} + C0$ |
| 0 | 1 | 0 | $F = R + \overline{S} + C0$ |
| 0 | 1 | 1 | $F = R \vee S$ |
| 1 | 0 | 0 | $F = R \wedge S$ |
| 1 | 0 | 1 | $F = \overline{R} \wedge S$ |
| 1 | 1 | 0 | $F = R \oplus S$ |
| 1 | 1 | 1 | $F = \overline{(R \oplus S)}$ |

Арифметичні операції АЛП виконуються з використанням чисел R і S, які можуть розглядатися як додаткові коди з урахуванням стану вхідного переноса **C0**. При цьому на виході **C4** формується стан вихідного переноса. Сигнал переповнення **OVR** при виконанні арифметичних операцій визначається з формули:

$$\mathbf{OVR = C4 \oplus C3,} \qquad (Д1.1)$$

де **C3** – значення сигналу вхідного переносу в старший (третій) розряд АЛП старшої МПС. Одиниця на виході OVR світчить, що результат операції перевищив розрядність цифрової частини і цифра займає знаковий розряд старшої МПС.

При нульовому значенні на шині **F** АЛП формує знак нульового результату (при цьому сигнал **Z** приймає значення одиниці).

***Регістр-акумулятор PQ*** звичайно використовується при виконанні "складних" операцій (наприклад, множення або ділення чисел). Регістр побудований на DC – тригерах із спрацьовуванням по передньому фронту тактового сигналу **CLK (0 → 1)**. При наявності умови дозволу запису (I8 I7 I6 = 000, 100 або 110) з приходом чергового фронту сигналу **CLK** у регістр записується інформація з виходу мультиплексора MXQ (див. рис. Д1.1), який в залежності від коду інструкції **I8 – I6**, формується як вихід АЛП (**F**), або як вихід зсувача ЗСВQ (**CQ**).

***Зсувач ЗСВQ*** виконує логічний зсув інформації ліворуч або праворуч на один розряд, інакше передає інформацію з виходу **Q** регістра PQ без зсуву. Якщо зсув не виконується, клеми **PQ3** і **PQ0** відключені від схеми і знаходяться в третьому стані. Робота цього зсувача аналогічна роботі зсувача ЗСВF (див. рис. Д1.3).

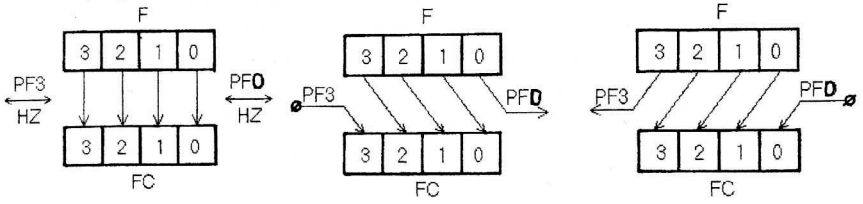
Адреса приймача результату АЛП задається кодом (I8 – I6) МК (табл.Д1.3).

Кодування приймача результата роботи АЛП. Таблиця Д1.3

| $I_8$ | $I_7$ | $I_6$ | Зсувач ЗСВF | Завантаження РОН | Зсувач ЗСВQ | Завантаження RQ | Вихід Y |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | - | - | - | F → RQ | F |
| 0 | 0 | 1 | - | - | - | - | F |
| 0 | 1 | 0 | - | F→ R(AB) | - | - | PA |
| 0 | 1 | 1 | - | F→ R(AB) | - | - | F |
| 1 | 0 | 0 | Праворуч | F/2 → R(AB) | Праворуч | RQ/2→RQ | F |
| 1 | 0 | 1 | Праворуч | F/2 → R(AB) | - | - | F |
| 1 | 1 | 0 | Ліворуч | 2 F → R(AB) | Ліворуч | 2RQ → RQ | F |
| 1 | 1 | 1 | Ліворуч | 2 F → R(AB) | - | - | F |

***Мультиплексор вихідних даних*** MXY, в залежності від коду інструкції **I8 –I6**, вибирає дані з виходу АЛП **F** або з виходу регістра PA.

***Шинний формувач ШФ*** з трьома станами керує виходом **Y** МІС. При низькому рівні сигналу ($\overline{OE}$) = 0 на виході **Y** з'являється стан шини **FY** мультиплексора MXY. Якщо сигнал ($\overline{OE}$) = 1, вихід **Y** відключається від схеми і знаходиться в третьому стані.




а) б) в)

Рисунок Д1.3. Работа зсувача ЗСВF:
а) - без зсуву ; б) – зсув праворуч; в) – зсув ліворуч

**Визначення вимог до часових параметрів МПС**

Припустимо, що адреса АА (АВ), а також інструкції **I** МПС одночасно надходять з регістра мікрокоманд (РМК). Нехай, наприклад, РМК виконаний на DC – тригерах зі спрацьовуванням по передньому фронті тактового сигналу **CLK (0→1)**. Тоді в моменти часу 1 і 3 (див. рис. Д1.4) після спрацьовування РМК на вході МПС будуть змінюватися коди чергових мікрокоманд: МК(i), МК(i+1), МК(i+2).

Такт роботи **T** процесорної секції ВС1 при обробці однієї мікрокоманди становить:

$$T = t_{12} + t_{23}, \qquad (Д1.2)$$

де $t_{12} \geq \max\{(t^* + t_{ЗАТ}^{ЗСВF}),\ (t^* + t_{ЗАТ}^{MXQ} + t_{ЗАТ}^{PQ})\};$

$t_{23} \geq \max\{t_{під}^{РМК},\ t_{під}^{PQ},\ t_{ЗАП}^{РЗП}\};$

$t^* \geq t_{ЗАТ}^{РМК} + t_{ЧІТ}^{РЗП} + t_{ЗАП}^{РА} + t_{ЗАТ}^{MXR} + t_{ЗАТ}^{АЛП};$

$t_{під}^{РМК}$, $t_{під}^{PQ}$ - час підготовки до спрацьовування DC – тригерів РМК і PQ відповідно;

$t_{ЧІТ}^{РЗП}$, $t_{ЗАП}^{РЗП}$ - час читання і запису інформації у регістрову пам'ять відповідно;

$t_{ЗАТ}^{РМК}$, $t_{ЗАТ}^{PQ}$ - час затримки РМК і PQ відповідно;

$t_{ЗАТ}^{MXQ}$, $t_{ЗАТ}^{MXR}$ - час затримки мультиплексорів MXQ і MXR відповідно;

$t_{ЗАТ}^{ЗСВF}$, $t_{ЗАТ}^{АЛП}$ - час затримки зсувача ЗСВF і АЛП відповідно.

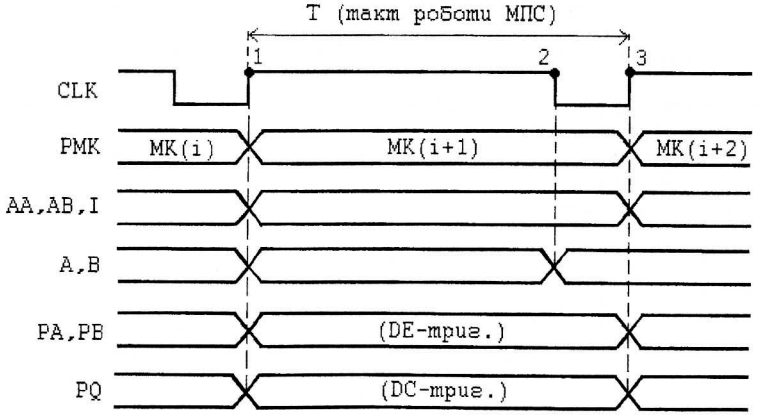



Рисунок Д1.4. Часова діаграма одного такту роботи МПС

## Д1.2 Організація блока обробки даних

Мікропроцесорні секції ВС1 можуть бути об'єднані з метою реалізації блока обробки даних (БОД) з розрядністю, кратної розрядності однієї МПС, тобто 4, 8, 12, ... На рис. Д1.5 показані зовнішні ланки БОД на **4n** розряди з **n** секцій ВС1: ВхП, ВихП – ланки вхідного і вихідного переносів відповідно; N – знак (розряд F3 старшої МПС); ПП – ознака переповнення; Z – ознака нульового результату на виході АЛП ВС1.

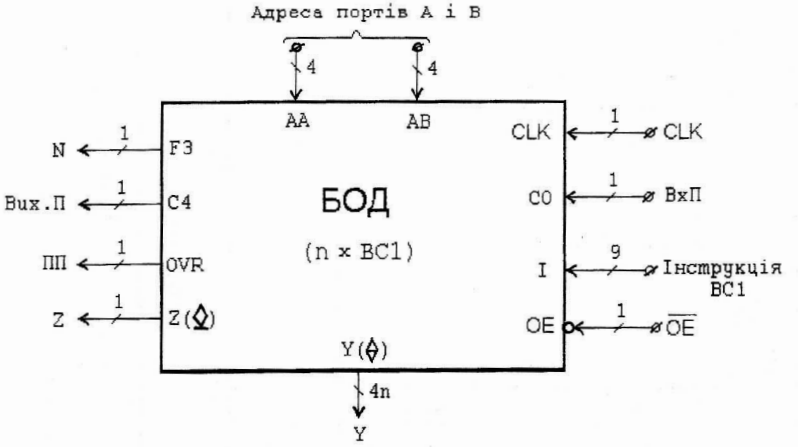



Рисунок Д1.5. Умовна позначка БОД з n секцій ВС1

Як приклад на рис. Д1.6 наведено 16 – розрядний БОД з використанням і без використання (пунктирні лінії) схеми прискореного переносу (СПП) на основі мікросхеми К1804ВР1. При відсутності СПП перенос у БОД при виконанні арифметичних операцій поширюється послідовно від входу до виходу. Такт роботи БОД T (Д1.2) буде збільшений за рахунок того, що

$$t_{ЗАТ}^{АЛП} = n \cdot t_{ЗАТ}^{ПЕР}, \qquad (Д1.3)$$

де $t_{ЗАТ}^{ПЕР}$ - час затримки поширення перенесу в одній секції.

Чим більше об'єднано МПС, тим повільніше працює БОД. На практиці для зменшення часу поширення переносу використовують мікросхеми СПП. Одна СПП типу К1804ВР1 дозволяє організувати паралельні ланки переносу у БОД до 16 розрядів (див. рис. Д1.6). Для підключення мікросхеми ВР1 до МПС використовуються спеціальні виводи: $\overline{P}$ - умова поширення переносу і $\overline{G}$ - місцевий перенос (з тетради).

Функції СПП формуються таким чином:

$$P_0 = \rho_3 \cdot \rho_2 \cdot \rho_1 \cdot \rho_0 = \overline{P(3-0)} \text{ - умова розповсюдження переносу через МПС0 БОД,} \qquad (Д1.4)$$

де $\rho_3 = R_3 \vee S_3$; $\rho_2 = R_2 \vee S_2$; $\rho_1 = R_1 \vee S_1$; $\rho_0 = R_0 \vee S_0$;

$R_3, R_2, R_1, R_0$ - входи АЛП молодшої МПС0 БОД;

$S_3, S_2, S_1, S_0$ - входи АЛП молодшої МПС0 БОД;

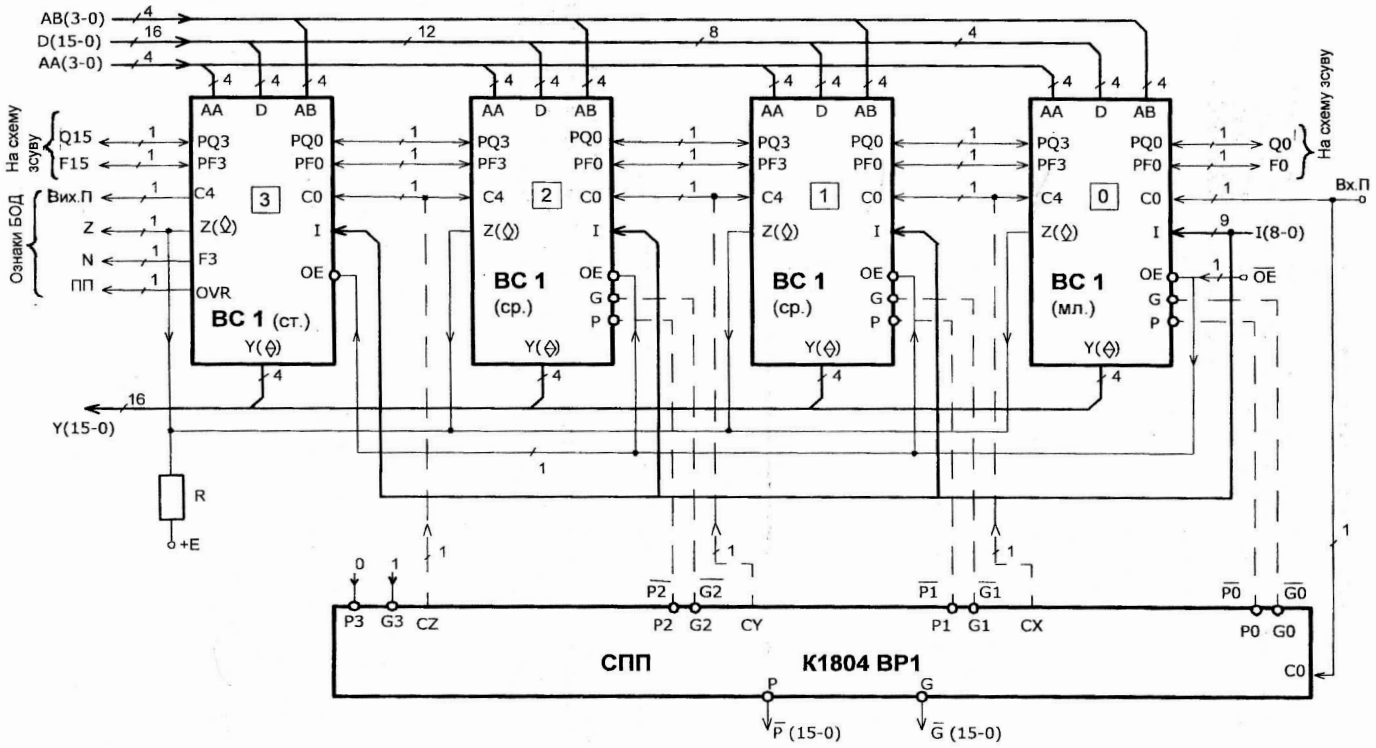



Рисунок Д1.6. Функціональна схема БОД на 16 розрядів

$\rho_3$, $\rho_2$, $\rho_1$, $\rho_0$ - умова розповсюдження переносу через розряди (3 - 0) АЛП молодшої секції БОД.

Аналогічно реалізовані сигнали $\overline{P_3}$, $\overline{P_2}$ та $\overline{P_1}$ відповідно в секціях МПС3, МПС2 та МПС1 БОД.

$$\overline{G_0} = \overline{g_3 + \rho_3 \cdot g_2 + \rho_3 \cdot \rho_2 \cdot g_1 + \rho_3 \cdot \rho_2 \cdot \rho_1 \cdot g_0} = \overline{G(3-0)} \text{ - місцевий перенос} \qquad (Д\,1.5)$$

секцій МПС0 із розрядів (3 - 0) до слідуючої тетради (7 - 0),

де $g_3$, $g_2$, $g_1$, $g_0$ - місцевий перенос з 3-го, 2-го, 1-го та 0-го розряду;

$g_3 = R_3 \wedge S_3$; $g_2 = R_2 \wedge S_2$; $g_1 = R_1 \wedge S_1$; $g_0 = R_0 \wedge S_0$;

Аналогічним чином формуються сигнали $\overline{G_3}$, $\overline{G_2}$ та $\overline{G_1}$ секцій МПС3, МПС2 та МПС1.

Групові функції в ВР1 формуються аналогічно:

$$\overline{P(15-0)} = \overline{P_3 \wedge P_2 \wedge P_1 \wedge P_0}; \qquad (Д\,1.6)$$

$$\overline{G(15-0)} = \overline{G_3 + P_3 \cdot G_2 + P_3 \cdot P_2 \cdot G_1 + P_3 \cdot P_2 \cdot P_1 \cdot G_0}. \qquad (Д\,1.7)$$

Переноси CZ, CY, та CX формуються згідно з формулами:

$$CX = G_0 + P_0 \cdot ВхП; \qquad (Д\,1.8)$$

$$CY = G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot ВхП; \qquad (Д\,1.9)$$

$$CZ = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot ВхП \qquad (Д\,1.10)$$

Схема прискореного переносу ВР1 зроблена таким чином, що вихідні сигнали CX, CY і CZ, які підключені відповідно на входи C0 у 1, 2 і 3 МПС (див. рис. Д1.6) формуються приблизно одночасно після затримки $t_{ЗАТ}^{СПП} = 10нс$ відносно вхідних сигналів підготовчих функцій. Вихідний перенос ВихП БОД формується не схемою ВР1, а старшої МПС з затримкою $t_{ЗАТ}^{C4-C0}$ (відносно вхідного переносу секції C0).

Зсуви в МПС можливо організувати як одинарної, так подвійної довжини (табл.Д1.4). При виконанні операцій зсуву необхідно за допомогою зовнішніх схем обрамлення ВС1 організувати формування інформації на двонаправлених шинах граничних МПС. Це можливо зробити за допомогою мультиплексорів (МХ) з тристабільними виходами. На рис. Д1.7 наведена функціональна схема організації зсуву при використанні (МХ) на основі мікросхеми К531КП11 [ 3 ]. Один такий МХ являє собою два комутатори чотирьох інформаційних входів D3 – D0 на один вихід, який може приймати три стійких стани. У залежності від стану керуючих входів A0 і A1 при формуванні на вході $\overline{OE} = 0$ сигнал з одного з обраного інформаційного входу передається на вихід. При подачі на керуючий вхід $\overline{OE}$ рівня логічної одиниці МХ переходить у третій стан незалежно від стану на його інших входах.

Керуючі сигнали $\overline{R}$ і $\overline{L}$ визначають МХ, який знаходиться в робочому (активному) стані. При цьому інші МХ знаходяться в третьому стані. Сьомий розряд (I7) інструкції ВС1 може бути використаний для визначення напрямку зсуву: I7 = 0 - зсув праворуч; I7 = 1 - зсув ліворуч. Розряди S1 і S0, як правило, формуються РМК і загалом з розрядом I7 мікрокоманди керують комутацією входів схем зсуву інформації.

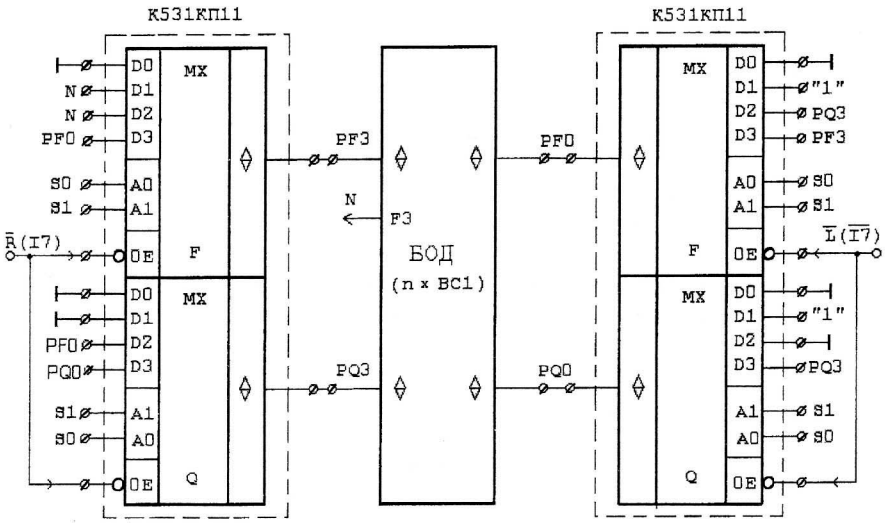



Рисунок Д1.7. Приклад організації схем зсуву на основі МХ КП11 (К531)

### Д1.3. Мікропроцесорна секція К1804ВС2

Мікропроцесорна секція К1804ВС2 (надалі ВС2) по призначенню і принципам побудови аналогічна ВІС К1804ВС1, крім наступних особливостей:

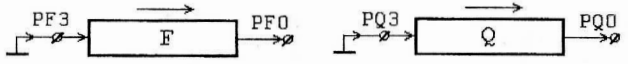
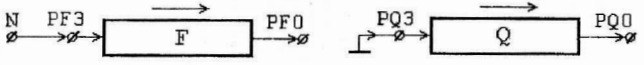
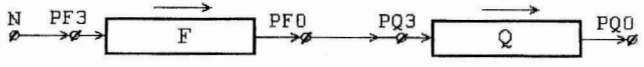
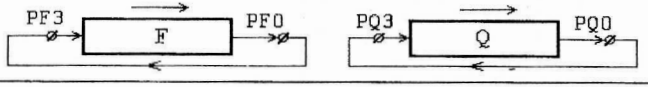
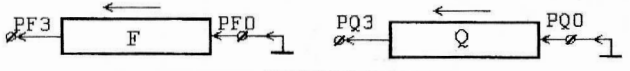
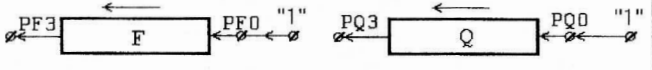
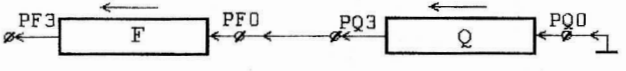
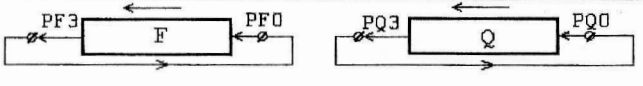
- наявність АЛП, який виконує не тільки арифметичні і булеві, але і спеціальні функції;
- наявність зсувача даних АЛП, який виконує операції логічного та арифметичного зсуву;
- наявність убудованих схем для програмно – апаратної реалізації алгоритмів множення, ділення, нормалізації, перетворення числа в додатковий код та деякі інші операції;
- наявність двох додаткових шин введення – виведення інформації;

Умовна функціональна схема МПС ВС2 наведена на рис. Д1.8. Регістровий запам'ятовуючий пристрій (РЗП) на 16 чотирьох розрядних слів реалізований на прозорих DE – тригерах типу "защілка". На відміну від РЗП МПС ВС1 він має вхід дозвілу запису інформації $\overline{WE}$:

$$\overline{WE} = \begin{cases} 0 \text{ - дозвіл запису даних з шини } Y(3-0) \text{ в РЗП;} \\ 1 \text{ – запис інформації в РЗП забороняється.} \end{cases}$$

Запис результату з шини Y(3-0) може відбуватися тільки через канал порту АВ протягом низького рівня тактового сигналу синхронізації CLK і при наявності активного (низького) рівня сигналу $\overline{WE}$ ( $\overline{WE} = 0$ ).

Можливі варіанти організації зсуву шини F та PQ на ВС1. Таблиця Д1.4

| I7 | S1 | S0 | Схема комутації розрядів зсування | Примітки |
|---|---|---|---|---|
| O(R) | 0 | 0 |  | Одинарий зсув вправо с занесенням нуля на PF3 |
| O(R) | 0 | 1 |  | Одинарий зсув вправо с занесенням знака N на PF3 |
| O(R) | 1 | 0 |  | Подвійний зсув вправо с занесенням знака N на PF3 |
| O(R) | 1 | 1 |  | Одинарний циклічний зсув вправо |
| 1(L) | 0 | 0 |  | Одинарий зсув вліво с занесенням нуля на PF3 |
| 1(L) | 0 | 1 |  | Одинарий зсув вліво с занесенням знака N на PF3 |
| 1(L) | 1 | 0 |  | Подвійний зсув вліво с занесенням знака N на PF3 |
| 1(L) | 1 | 1 |  | Одинарний циклічний зсув вліво |

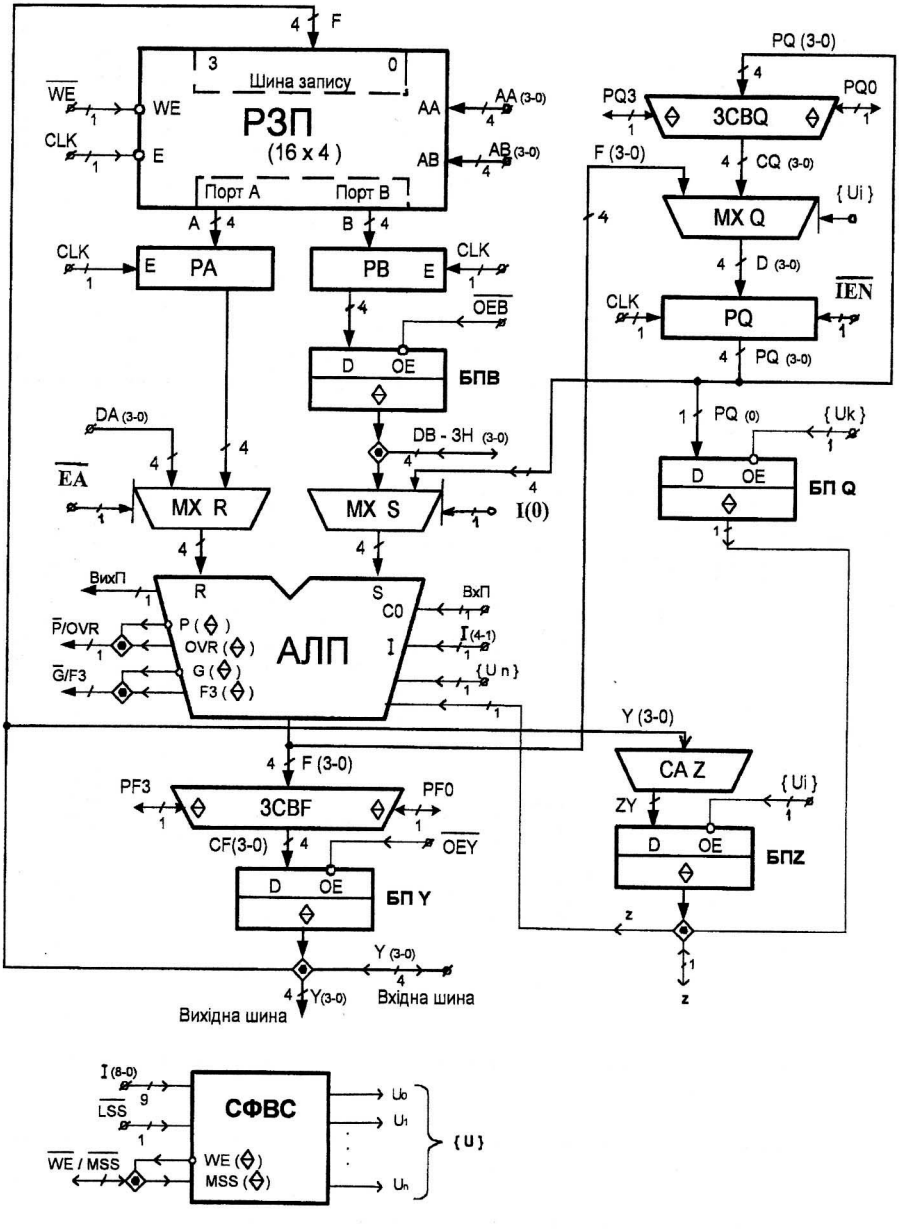



Рисунок Д1.8. Функціональна схема процесорного елемента К1804ВС2

З виходів портів A и B РЗП під дією високого рівня сигналу CLK інформація записується в буферні регістри PA і PB (виконані на DE – тригерах). На виході регістра PB, на відміну від регістра PA, підключено трьохстабільний буферний підсилювач БПВ керований сигналом $\overline{OEB}$. Якщо сигнал $\overline{OEB} = 1$, вихід БПВ знаходиться в третьому стані (у такий спосіб вихід регістра PB від'єднується від шини DB). У цьому випадку на шину DB(3 – 0) може бути подана інформація з зовнішнього джерела даних через зовнішню шину 3DB(3 – 0). Двонапрямкий режим роботи шини 3DB визначає характер елементів, які можуть бути підключені до цієї шини.

Комутатори MXR і MXS здійснюють вибір джерел на шини R і S АЛП. Шина S(3 – 0) безпосередньо керується молодшим бітом коду інструкції I0:

$$S = \begin{cases} DB, & \text{якщо } I0 = 0; \\ PQ, & \text{якщо } I0 = 1. \end{cases}$$

Шина R керується сигналом $\overline{EA}$:

$$R = \begin{cases} PA, & \text{якщо } \overline{EA} = 0; \\ DA\,(\text{шина введення даних}), & \text{якщо } \overline{EA} = 1. \end{cases}$$

В цілому вибір джерел операндів АЛП наведено в табл. Д1.5.

Джерела операндів АЛП. Таблиця Д1.5

| $\overline{EA}$ | I0 | $\overline{OEB}$ | R | S |
|---|---|---|---|---|
| 0 | 0 | 0 | PA | PB |
| 0 | 0 | 1 | PA | 3DB |
| 0 | 1 | * | PA | PQ |
| 1 | 0 | 0 | DA | PB |
| 1 | 0 | 1 | DA | 3DB |
| 1 | 1 | * | DA | PQ |

Блок АЛП забезпечує виконання 7 арифметичних і 9 логічних операцій, а також 9 спеціальних функцій над одним або двома операндами R і S, які надходять з виходів комутаторів.

В МПС ВС2 сигнали $\overline{P}$ (умова поширення переносу) і OVR (переповнення додаткових кодів), а також $\overline{G}$ (сигнал місцевого переносу) і F3 (старший розряд секції) керуються сигналами $\overline{MSS}$ і $\overline{LSS}$. В БОД у молодшій і середній МПС використовуються на двофункціональних клемах та формуються сигнали $\overline{P}$ і $\overline{G}$ відповідно (для схеми прискореного переносу), а в старшій МПС - сигнали OVR і F3 відповідно. Клеми вводу переносу (C0) і виводу переносу (C4) в кожній МПС забезпечують поширення послідовного переносу якщо використовується послідовна схема організації переносу.

Формування операції АЛП виконується сигналами ( I8 – I0 ) інструкції. Якщо на входи (I4 – I0) поданий сигнал логічного нуля (низький рівень), АЛП виконує спеціальні функції, тип яких задається кодом інструкції ( I8 – I5 ). Якщо сигнали (I4 – I0) ≠ 0, виконується одна з 16 операцій, які наведені в табл. Д1.6.

Розглянемо, наприклад, виконання операції віднімання в додатковому коді інструкції (I4 – I1) = 0001 → ( F = S + R + C0 ), якщо R = (- 5 ) дк = 1. 011, S = (- 6 ) дк = 1. 010. Очевидно, в цьому разі на шині F буде створений результат:

| | |
|---|---|
| 0. 100 | $\overline{R}$ |
| 1. 010 | $\overline{S}$ |
| 1 | C0 |
| 1.111 | F = (- 1)дк |

Таким чином, якщо сигнал C0 = 1, при (I4 – I1) = 0001 виконується операція віднімання (S – R) і результат формується в додатковому коді.

Регістр – акумулятор PQ складають чотири непрозорі DC – тригери. Запис інформації в PQ відбувається по передньому фронту тактового сигналу CLK при низькому рівні сигналу дозволу запису $\overline{IEN}$ = 0. Якщо $\overline{IEN}$ = 1, здійснюється режим збереження інформації і CLK блокується.

Зсувач ЗСВQ виконує логічний зсув на один розряд ліворуч або праворуч, або передачу інформації (з виходу F АЛП або виходу PQ) без зсуву. В останньому випадку виводи PQ3 та PQ0 знаходяться в третьому (відключеному) стані. Схема зсувача ЗСВQ працює аналогічно відповідним схемам секції ВС1 (див. рис. Д1.3).

Операції АЛП ВС2. Таблиця Д1.6

| $I_4$ | $I_3$ | $I_2$ | $I_1$ | **Функція АЛП** |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 1 1 1 |
| 0 | 0 | 0 | 1 | $S+\overline{R}+C0$ |
| 0 | 0 | 1 | 0 | $R+\overline{S}+C0$ |
| 0 | 0 | 1 | 1 | R + S + C0 |
| 0 | 1 | 0 | 0 | S + C0 |
| 0 | 1 | 0 | 1 | $\overline{S}+C0$ |
| 0 | 1 | 1 | 0 | R + C0 |
| 0 | 1 | 1 | 1 | $\overline{R}+C0$ |
| 1 | 0 | 0 | 0 | 0 0 0 0 |
| 1 | 0 | 0 | 1 | $\overline{R}+S$ |
| 1 | 0 | 1 | 0 | $\overline{R \oplus S}$ |
| 1 | 0 | 1 | 1 | R ⊕ S |
| 1 | 1 | 0 | 0 | R ∧ S |
| 1 | 1 | 0 | 1 | $\overline{R \vee S}$ |
| 1 | 1 | 1 | 0 | $\overline{R \wedge S}$ |
| 1 | 1 | 1 | 1 | R ∨ S |

Зсувач АЛП ЗСВF, на відміну від зсувача ЗСВQ, крім логічних зсувів (коли всі розряди зсуваються однаково), може виконувати також арифметичний зсув (старший знаковий розряд передається без зсуву). Схема арифметичного зсуву ліворуч та праворуч наведена на рис. Д1.9.

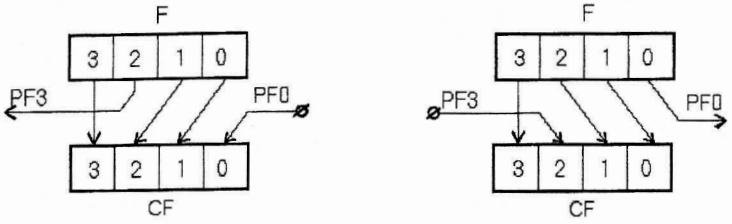



Рисунок Д1.9. Робота зсувача ЗСЧF при виконанні арифметичного зсуву

Арифметичний зсув ліворуч здійснюється коректно, якщо знаковий розряд F3 і спадаючий при зсуві розряд PF3 старшої МПС утворюють коди 00 або 11. В іншому випадку необхідно формувати сигнал помилки.

Вихід зсувача ЗСВF з'єднанний з буферним підсилювачем БПУ, який налагоджується сигналом $\overline{OEY}$. При низькому рівні сигналу ($\overline{OEY} = 0$) інформація з виходу зсувача передається на вихідну шину Y(3 – 0), а також надходить на вхід РЗП для запису. Якщо сигнал $\overline{OEY} = 1$, виходи БПУ перебувають в третьому стані і шину Y (3 – 0) можна використовувати для запису інформації в РЗП від зовнішнього (по відношенню до ВС2) джерела даних.

Формувач ознаки нуля (CAZ) формує сигнал Z = 1 у тому випадку, коли всі сигнали на виході АЛП мають нульові стани. При виконанні деяких спеціальних функцій шина Z використовується як вхід в ВС2.

Вибір приймача результату (табл. Д1.7) і керування зсувачами PQ та АЛП здійснюється сигналами (I8 – I5) коду інструкції ВС2.

Вибір в ВС2 приймача результату. Таблиця Д1.7

| $I_8$ | $I_7$ | $I_6$ | $I_5$ | Функція ЗСВF | $\overline{WE}$ | Функція ЗСВQ |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | (АЗ) F / 2 → DY | 0 | Збереження |
| 0 | 0 | 0 | 1 | (ЛЗ) F / 2 → DY | 0 | Збереження |
| 0 | 0 | 1 | 0 | (АЗ) F / 2 → DY | 0 | (ЛЗ) PQ / 2 → PQ |
| 0 | 0 | 1 | 1 | (ЛЗ) F / 2 → DY | 0 | (ЛЗ) PQ / 2 → PQ |
| 0 | 1 | 0 | 0 | F → DY | 0 | Збереження |
| 0 | 1 | 0 | 1 | F → DY | 1 | (ЛЗ) PQ / 2 → PQ |
| 0 | 1 | 1 | 0 | F → DY | 1 | F → PQ |
| 0 | 1 | 1 | 1 | F → DY | 0 | F → PQ |
| 1 | 0 | 0 | 0 | (АЗ) 2·F → DY | 0 | Збереження |
| 1 | 0 | 0 | 1 | (ЛЗ) 2·F → DY | 0 | Збереження |
| 1 | 0 | 1 | 0 | (АЗ) 2·F → DY | 0 | (ЛЗ) 2·PQ → PQ |

Продовження таблиці Д1.7

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | (ЛЗ) 2 F → DY | 0 | (ЛЗ) 2 PQ → PQ |
| 1 | 1 | 0 | 0 | F → DY | 1 | Збереження |
| 1 | 1 | 0 | 1 | F → DY | 1 | (ЛЗ) 2 PQ → PQ |
| 1 | 1 | 1 | 0 | PF0 → DY (0 - 3 ) | 0 | Збереження |
| 1 | 1 | 1 | 1 | F → DY | 0 | Збереження |

**Примітки:** ЛЗ – логічний зсув; АЗ - арифметичний зсув; $\overline{WE}$ – сигнал генерується на клемі $\overline{M}$ молодшої секції БОД кодом (I8 – I5).

Крім стандартних операцій МПС ВС2 можуть також виконувати спеціальні функції (табл. Д1.8): множення без знака по алгоритму "А", множення додаткових кодів по алгоритму "А", ділення додаткових кодів по алгоритму "а", нормалізації чисел і деякі інші функції. Докладні описи алгоритмів виконання спеціальних функцій ВС2 можливо знайти, наприклад, у роботах [1, 2].

## Д1.4 Блок обробки даних на основі ВС2

При з'єднанні декількох МПС ВС2 кожну секцію необхідно налоготити як молодшу, середнью або старшу. Для цього використовуються лінії LSS і MSS / $\overline{W}$ (табл. Д1.9).

Налогодження позиції МПС К1804ВС2 в БОД. Таблиця Д1.9

| Позиція процесорної секції | $\overline{LSS}$ | $\overline{MSS}$ / $\overline{W}$ |
|---|---|---|
| Молодша | "0" ( вхід ) | $\overline{W}$ ( вихід ) |
| Середня | "1" ( вхід ) | "1" ( вхід ) |
| Старша | "1" ( вхід ) | "0" ( вхід ) |

**Примітка:** $\overline{W}$ - дозвіл запису в РЗП ВС2 в поточному такті.

Формування активного рівня стробу запису $\overline{W}$ забороняється деякими інструкціями (див. табл. Д1.7). В більшості випадках (наприклад, при I8 – I5 = 0100) сигнал $\overline{W}$ = 0 формується в продовж поточного такту роботи ВС2. Це дає можливість за допомогою коду операції ВС2 блокувати запис інформації в РЗП ВС2 по адресу АВ. Для цього режиму необхідно за допомогою зовнішньої комутації з'єднати клему MSS / $\overline{W}$ молодшої секції з клемами $\overline{WE}$ молодшої, середніх і старшої МПС ВС2 (рис. Д1.10).

Спеціальні функції АЛП ( I0 – I4 =0; $\overline{IEN}$ = 0) ВС2. Таблиця Д1.8

| $I_8$ | $I_7$ | $I_6$ | $I_5$ | **Функція АЛП** | **Операція АЛП ( при Z = 0)** | **Операція АЛП ( при Z = 1)** | **Зсувач ЗСВF** | **Зсувач ЗСВQ** |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Множення без знаку (алг. "А") | F = S + C0 | F = R + S + C0 | F/2→Y (ЛЗ) | PQ / 2→PQ (ЛЗ) |
| 0 | 0 | 1 | 0 | Множення додатков. кодів (алгоритм "А") | F = S + C0 | F = R + S + C0 | F/2→Y PF3 = F3⊕ OVR | PQ/2→PQ (ЛЗ) |
| 0 | 1 | 0 | 0 | Інкремент на 1 або 2 | F = S + 1 + C0 | F = S + 1 + C0 | F → Y | PQ → PQ |
| 0 | 1 | 0 | 1 | Перетворення у додатковий код | F = S + C0 | F = $\overline{S}$ + C0 | F → Y | PQ → PQ |
| 0 | 1 | 1 | 0 | Крок корекції множення (на знак ) у додатковому коді | F = S + C0 | F = S + $\overline{R}$ + C0 | F/2→Y PF3 = F3⊕ OVR | PQ/2→PQ (ЛЗ) |
| 1 | 0 | 0 | 0 | Нормалізація слова одиночної довжини | F = S + C0 | F = S + C0 | F → Y | 2PQ→PQ (ЛЗ) |
| 1 | 0 | 1 | 0 | Нормалізація слова подвійної довжини | F = S + C0 | F = S + C0 | 2F→Y (ЛЗ) | 2PQ→ PQ (ЛЗ) |
| 1 | 1 | 0 | 0 | Ділення (алг. "а") додаткових кодів | F = S + R +C0 | F = S + $\overline{R}$ + C0 | 2F→ Y (ЛЗ) | 2PQ → PQ (ЛЗ) |
| 1 | 1 | 1 | 0 | Крок корецїї ділення додаткових кодів | F = S + R +C0 ( + НПР) | F = S + $\overline{R}$ + C0 ( + ПР ) | F → Y | 2PQ → PQ (ЛЗ) |

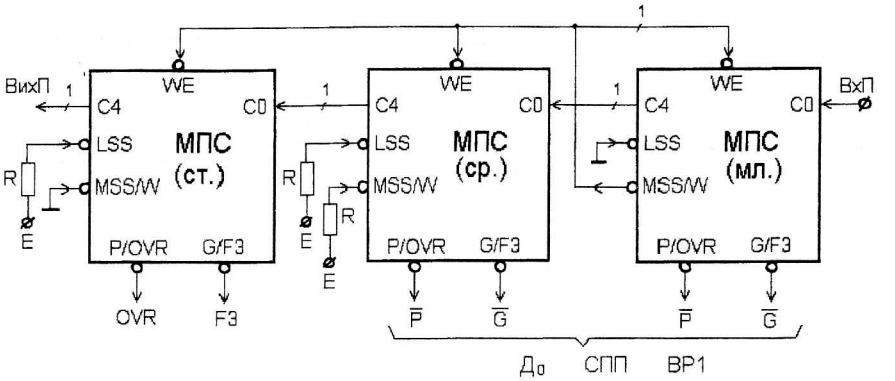



Рисунок Д1.10. Стандартна схема комутації в БОД секцій ВС2

Сигнал дозволу запису $\overline{WE}$ можливо також формувати в пристрою керування. Для цього у форматі мікрокоманди необхідно використати один розряд для сигналу $\overline{WE}$.

На рис. Д1.10 наведена схема стандартної комутації МПС ВС2 ( молодшої, середньої і старшої секції). Сигнал переповнення OVR формується в тому випадку, коли знакові розряди входів R і S "+", а знаковий розряд результату F "-", або навпаки. Фіксація цих випадків в ВС2 визначається порівнянням вхідного С3 і вихідного С4 переносів в знаковому розряді АЛП ВС2:

$$\mathbf{OVR = C4 \oplus C3} \qquad (Д1.5)$$

В якості приклада розглянемо виконання операції нормалізації числа у додатковому коді в БОД на основі МПС ВС2 [1, 2]. Ця операція виконується шляхом зсуву числа ліворуч. Зсуви припиняються, коли два старших розряди будуть мати різні значення. Приклади виконання операції нормалізації чисел наведені в табл. Д1.10.

Виконання нормалізації чисел у додатковому коді. Таблиця Д1.10

| Число до нормалізації | Число після нормалізації |
|---|---|
| 0000 0010 1100 0111 | 0101 1000 1110 0000 |
| 1110 1011 0100 0101 | 1010 1101 0001 0100 |
| 0000 0000 0000 0000 | Операція не виконується |
| 0110 1011 0101 1010 | 0110 1011 0101 1010 |
| 1111 1111 1111 1111 | 1000 0000 0000 0000 |

Умовні графічні позначення ВІС обробки даних наведені на рис. Д1.11.

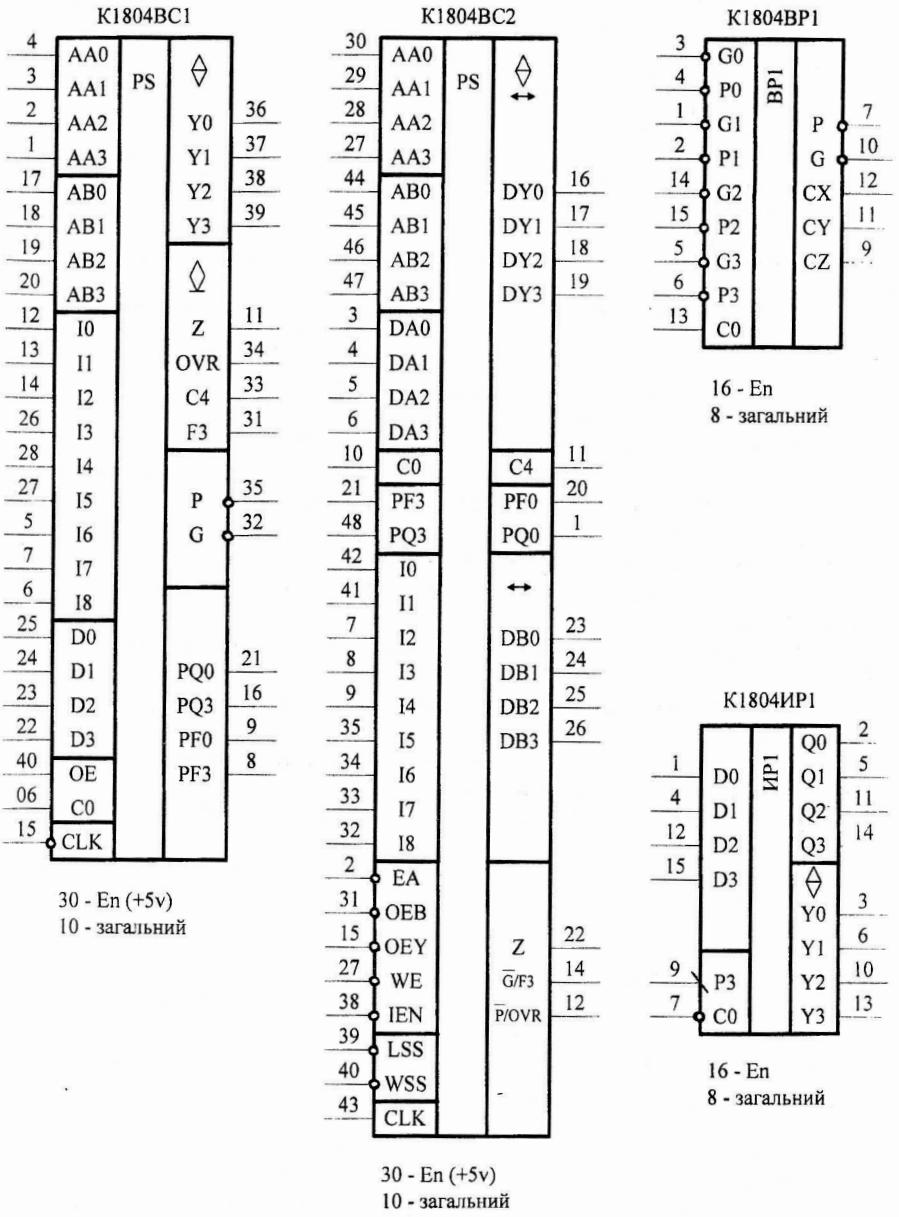



Рисунок Д1.11. Умовні графічні позначки ВІС обробки даних

## Д2. МЕТОДИЧНІ ВКАЗІВКИ ДО ПРОЕКТУВАННЯ ТА СИМУЛЯЦІЇ БЛОКІВ МІКРОПРОГРАМНОГО КЕРУВАННЯ

### Д2.1. Мікропроцесорна секція керування адресою мікрокоманди К1804ВУ4

Мікросхема К1804ВУ4 (надалі ВУ4) призначена для реалізації блоків мікропрограмного керування [1, 2]. Основна функція ВУ4 полягає у формуванні 12 – розрядних адрес мікрокоманд (МК), які зберігаються в мікропрограмної пам'яті (МПП).

Функціональна схема ВІС ВУ4 наведена на рис. Д2.1, де прийняті наступні позначення сигналів:

**MI (3 – 0)** - вхідна шина інструкції мікрокоманди. Визначає одну з 16 інструкцій ВУ4.

**DA (11 – 0)** - код зовнішньої шини адреси. Використовується як одне з чотирьох джерел адреси наступної мікрокоманди і як інформаційний вхід при запису інформації в РА / СТ.

$\overline{\textbf{CC}}$ - вхід сигналу умови, яка перевіряється.

**CCE** - вхід дозволу перевірки сигналу умови.

**C0** - вхід інкрементації лічильника мікрокоманд (ЛМК).

$\overline{\textbf{OEY}}$ - вхід дозволу видачі адреси на вихідну шину **Y (11 – 0)**. Використовується для відмикання буферного підсилювача ВУ4 з трьома станами (БПУ).

$\overline{\textbf{RLD}}$ - вхід дозволу запису інформації в **РА / СТ.**

**Y (11 – 0)** - вихідна шина адреси мікрокоманд.

$\overline{\textbf{FL}}$ - вихід ознаки переповнення стека. Використовується для індикації стану заповненого стека.

$\overline{\textbf{PE}}$ - вихідний сигнал дозволу подачі адреси на вхід **DA** з регістра мікрокоманд (РМК).

$\overline{\textbf{ME}}$ - вихідний сигнал дозволу подачі адреси на вхід **DA** з перетворювача початкової адреси.

$\overline{\textbf{VE}}$ - вихідний сигнал дозволу подачі коду адреси на вхід **DA** з виходу спеціального перетворювача векторної адреси.

***Мультиплексор адреси МХ*** на чотири входи для вибору джерел адреси наступної мікрокоманди:

- **РА / СТ** – внутрішній регістр адреси / лічильник;
- **РАМК (РС)** – лічильник мікрокоманд;
- **ST** - вихід стека;
- **DA** - код з зовнішньої шини адреси.

Обраний мультиплексором адрес **АМК** надходить на буферний підсилювач з трьома станами БПУ. Керування буферним підсилювачем здійснюється за допомогою зовнішнього керуючого сигналу $\overline{\textbf{OEY}}$ (звичайно подається з РМК). При низькому рівні сигналу ($\overline{\textbf{OEY}}$ = 0) БПУ дозволяє видачу адреси на вихідну шину **Y(11 - 0).** У противоположному випадку (при $\overline{\textbf{OEY}}$ = 1) вихід БПУ знаходиться в третьому (відключеному) стані.

***Регістр адреси / лічильник (РА / СТ)*** виконаний на 12 **DC** – тригерах, запис інформації в які здійснюється по позитивному перепаду ($0 \rightarrow 1$) тактового сигнала **CLK** при відповідній інструкції **MI**, або при наявності сигналу $\overline{\textbf{RLD}}$ = 0 поза залежністю від коду інструкції. Блок **РА / СТ** звичайно використовується як буфер для запису і збереження адреси, або коду кількості повторення циклу. У кожному потоковому такті вміст **РА / СТ** зменшується на одиницю. Якщо через шину **DA** у **РА / СТ** завантажується **N**, то при відповідній інструкції (наприклад, **RFCT**), цикл буде виконаний (**N** + 1) разів.

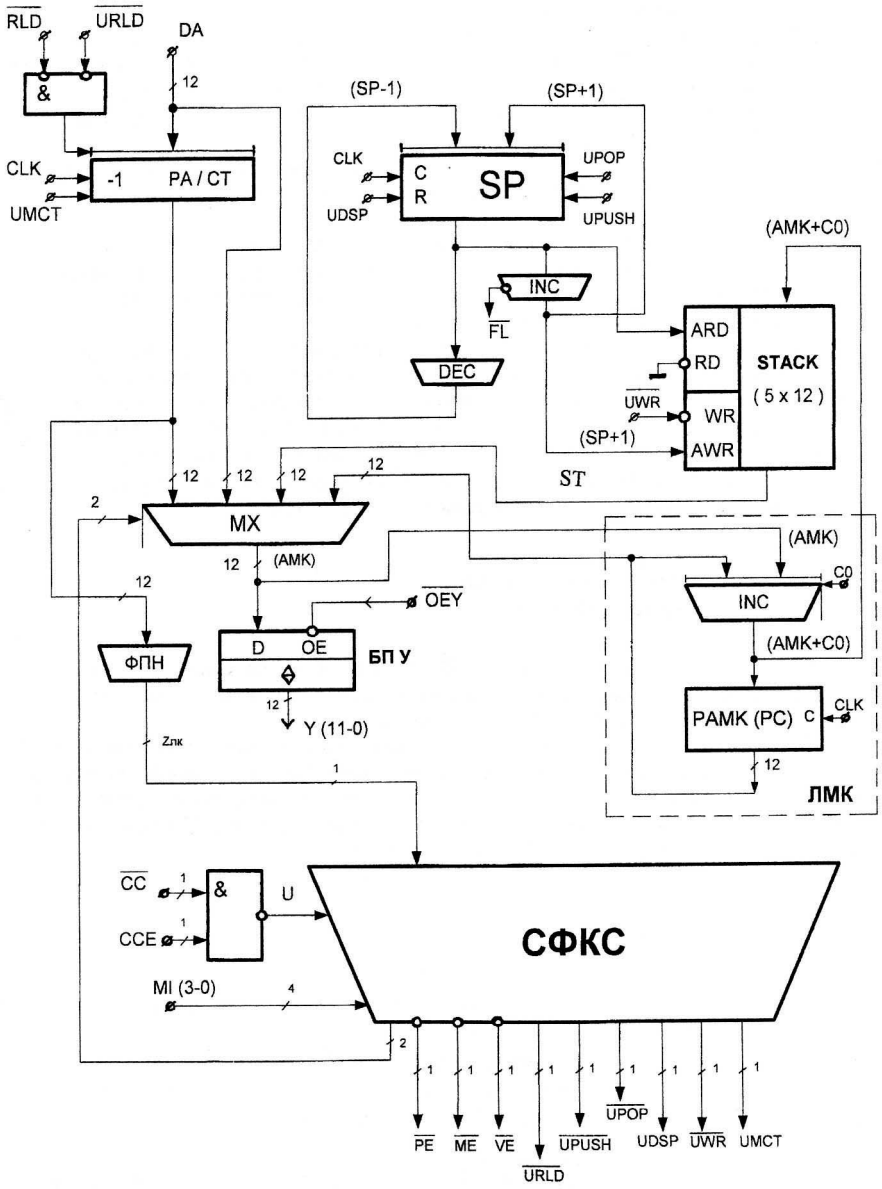



Рисунок Д2.1. Функціональна схема СКАМ К1804ВУ4

Система мікрокоманд мікросхеми К1804ВУ4. Таблиця Д2.1

| Мнем. познач. | МІ (3-0) | СТ стан. | Y (U=0) | Стек (U=0) | Y (U=1) | Стек (U=1) | СТ уст. | Вихідний сигнал |
|---|---|---|---|---|---|---|---|---|
| **JZ** | 0 0 0 0 | * | 0 | CLEAR | 0 | CLEAR | - | $\overline{PE}$ |
| **CJS** | 0 0 0 1 | * | PC | - | DA | PUSH | - | $\overline{PE}$ |
| **JMAP** | 0 0 1 0 | * | DA | - | DA | - | - | $\overline{ME}$ |
| **CJP** | 0 0 1 1 | * | PC | - | DA | - | - | $\overline{PE}$ |
| **PUSH** | 0 1 0 0 | * | PC | PUSH | PC | PUSH | 1) | $\overline{PE}$ |
| **JSRP** | 0 1 0 1 | * | CT | PUSH | DA | PUSH | - | $\overline{PE}$ |
| **CJV** | 0 1 1 0 | * | CT | - | DA | - | - | $\overline{VE}$ |
| **JRP** | 0 1 1 1 | * | CT | - | DA | - | - | $\overline{PE}$ |
| **RFCT** | 1 0 0 0 | ≠0 | ST | - | ST | - | DEC | $\overline{PE}$ |
| | | =0 | PC | POP | PC | POP | - | $\overline{PE}$ |
| **RPCT** | 1 0 0 1 | ≠0 | DA | - | DA | - | DEC | $\overline{PE}$ |
| | | =0 | PC | - | PC | - | - | $\overline{PE}$ |
| **CRTN** | 1 0 1 0 | * | PC | - | ST | POP | - | $\overline{PE}$ |
| **CJPP** | 1 0 1 1 | * | PC | - | DA | POP | - | $\overline{PE}$ |
| **LDCT** | 1 1 0 0 | * | PC | - | PC | - | | $\overline{PE}$ |
| **LOOP** | 1 1 0 1 | * | ST | - | PC | POP | - | $\overline{PE}$ |
| **CONT** | 1 1 1 0 | * | PC | - | PC | - | - | $\overline{PE}$ |
| **TWB** | 1 1 1 1 | ≠0 | ST | - | PC | POP | DEC | $\overline{PE}$ |
| | | =0 | DA | POP | PC | POP | - | $\overline{PE}$ |

**Примітки.** 1 ) - коли сигнали $\overline{CC}$ та CCE ≠ 0 виконується загрузка СТ, інакше СТ зберігає своє значення ; передбачається, що сигнал C0 = 1 ;

* - довільне значення.

**CLEAR** : SP : = 0 ; **LOAD** : $\overline{RLD}$ : = 0 , DA → CT

**PUSH** : SP + 1 → SP , PC → stack ( SP )

**POP** : ST : = stack (SP) , SP : = SP - 1

Вважається, що умова виконана якщо **U** = 1, тобто коли сигнал $\overline{\mathbf{CC}}$ = 0 і **CCE** = 1 (або CCE = 0 і $\overline{CC}$ = *). У цьому випадку виробляється дія, зазначена в названій операції. У противоположному випадку (якщо сигнал **U** = 0) реалізується протилежна дія (звичайно це інкрементування вмісту РАМК і адреса в МПП надходить з ЛМК). Незалежно від сигналу на вході $\overline{CC}$ можливо забезпечити виконання умови (**U** = 1), встановивши на вході **CCE** сигнал логічного нуля.

Блок **СФКС** виробляє також три керуючих сигнала $\overline{\mathbf{PE}}$, $\overline{\mathbf{ME}}$ та $\overline{\mathbf{VE}}$, які використовуються для відмикання одного з трьох буферних підсилювачів (БП) при підключенні зовнішніх джерел адреси до шини **DA** ВУ4 (рис. Д2.2).

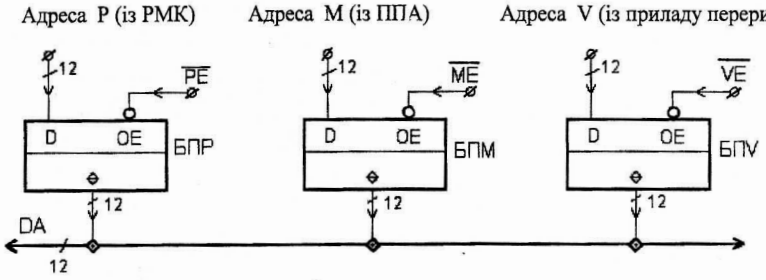



Рисунок Д.2.2. Вибір джерела адреси на шину DA ВУ4

Кожна інструкція ВУ4 виробляє тільки один активний сигнал низького рівня ($\overline{\mathbf{PE}}$, $\overline{\mathbf{ME}}$, $\overline{\mathbf{VE}}$), який відмикає свій БП і подає на шину **DA** відповідний адрес:

- від регістра мікрокоманд (при $\overline{\mathbf{PE}}$ = 0);
- від перетворювача початкової адреси ( при $\overline{\mathbf{ME}}$ = 0);
- від джерела векторної адреси при виникненні переривання (при $\overline{\mathbf{VE}}$ = 0).

Вихідна шина адреси **Y (11 – 0)** ВУ4 має три стани. Це надає можливість забезпечити зовнішній доступ до входів адреси МПП для передачі керування на початкову мікропрограму ініціалізації мікропроцесорної системи, або для організації тестової послідовності мікрокоманд.

У табл. Д2.1 наведені інструкції мікросхеми ВУ4, а на рис. Д2.3 - приклади виконання цих інструкцій. Розглянемо як приклад роботу окремих інструкцій.

***Інструкція CONT (MI = 1110) "Продовжити".*** По цій інструкції адреса наступної мікрокоманди визначається шляхом INC лічильника мікрокоманд (PC : = PC + 1). Використовується для організації послідовної вибірки мікрокоманд із МПП.

***Інструкція JZ (MI = 0000) "Перехід до нульової адреси".*** Виконує перехід до МК з нульовою адресою і відбувається очищення стека шляхом обнуління покажчика стека (SP : = 0).

***Інструкція CJP (MI = 0011) "Умовний перехід за адресою з РМК".*** При виконанні умови (**U** = 1) здійснюється перехід за адресою, яка приймається з РМК. При цьому у ВУ4 виробляється сигнал $\overline{\mathbf{PE}}$ = 0, який відмикає буферний підсилювач БПР (рис. Д2.2) і на шині **DA** з'являється адреса з РМК. Якщо умова не виконана (**U** = 0), то здійснюється перехід до наступної МК, адреса якої надходить з лічильника адреси мікрокоманд РС.

Виконання інших інструкцій пропонується розібрати самостійно.

Система мікрокоманд мікросхеми К1804ВУ4. Таблиця Д2.1

| Мнем. познач. | МІ (3-0) | СТ стан. | Y (U=0) | Стек (U=0) | Y (U=1) | Стек (U=1) | СТ уст. | Вихідний сигнал |
|---|---|---|---|---|---|---|---|---|
| **JZ** | 0 0 0 0 | * | 0 | CLEAR | 0 | CLEAR | - | $\overline{PE}$ |
| **CJS** | 0 0 0 1 | * | PC | - | DA | PUSH | - | $\overline{PE}$ |
| **JMAP** | 0 0 1 0 | * | DA | - | DA | - | - | $\overline{ME}$ |
| **CJP** | 0 0 1 1 | * | PC | - | DA | - | - | $\overline{PE}$ |
| **PUSH** | 0 1 0 0 | * | PC | PUSH | PC | PUSH | 1) | $\overline{PE}$ |
| **JSRP** | 0 1 0 1 | * | CT | PUSH | DA | PUSH | - | $\overline{PE}$ |
| **CJV** | 0 1 1 0 | * | CT | - | DA | - | - | $\overline{VE}$ |
| **JRP** | 0 1 1 1 | * | CT | - | DA | - | - | $\overline{PE}$ |
| **RFCT** | 1 0 0 0 | ≠0 | ST | - | ST | - | DEC | $\overline{PE}$ |
| | | =0 | PC | POP | PC | POP | - | $\overline{PE}$ |
| **RPCT** | 1 0 0 1 | ≠0 | DA | - | DA | - | DEC | $\overline{PE}$ |
| | | =0 | PC | - | PC | - | - | $\overline{PE}$ |
| **CRTN** | 1 0 1 0 | * | PC | - | ST | POP | - | $\overline{PE}$ |
| **CJPP** | 1 0 1 1 | * | PC | - | DA | POP | - | $\overline{PE}$ |
| **LDCT** | 1 1 0 0 | * | PC | - | PC | - | | $\overline{PE}$ |
| **LOOP** | 1 1 0 1 | * | ST | - | PC | POP | - | $\overline{PE}$ |
| **CONT** | 1 1 1 0 | * | PC | - | PC | - | - | $\overline{PE}$ |
| **TWB** | 1 1 1 1 | ≠0 | ST | - | PC | POP | DEC | $\overline{PE}$ |
| | | =0 | DA | POP | PC | POP | - | $\overline{PE}$ |

**Примітки.** 1 ) - коли сигнали $\overline{CC}$ та CCE ≠ 0 виконується загрузка СТ, інакше СТ зберігає своє значення ; передбачається, що сигнал C0 = 1 ;

* - довільне значення.

**CLEAR** : SP : = 0 ; **LOAD** : $\overline{RLD}$ : = 0 , DA → CT

**PUSH** : SP + 1 → SP , PC → stack ( SP )

**POP** : ST : = stack (SP) , SP : = SP - 1

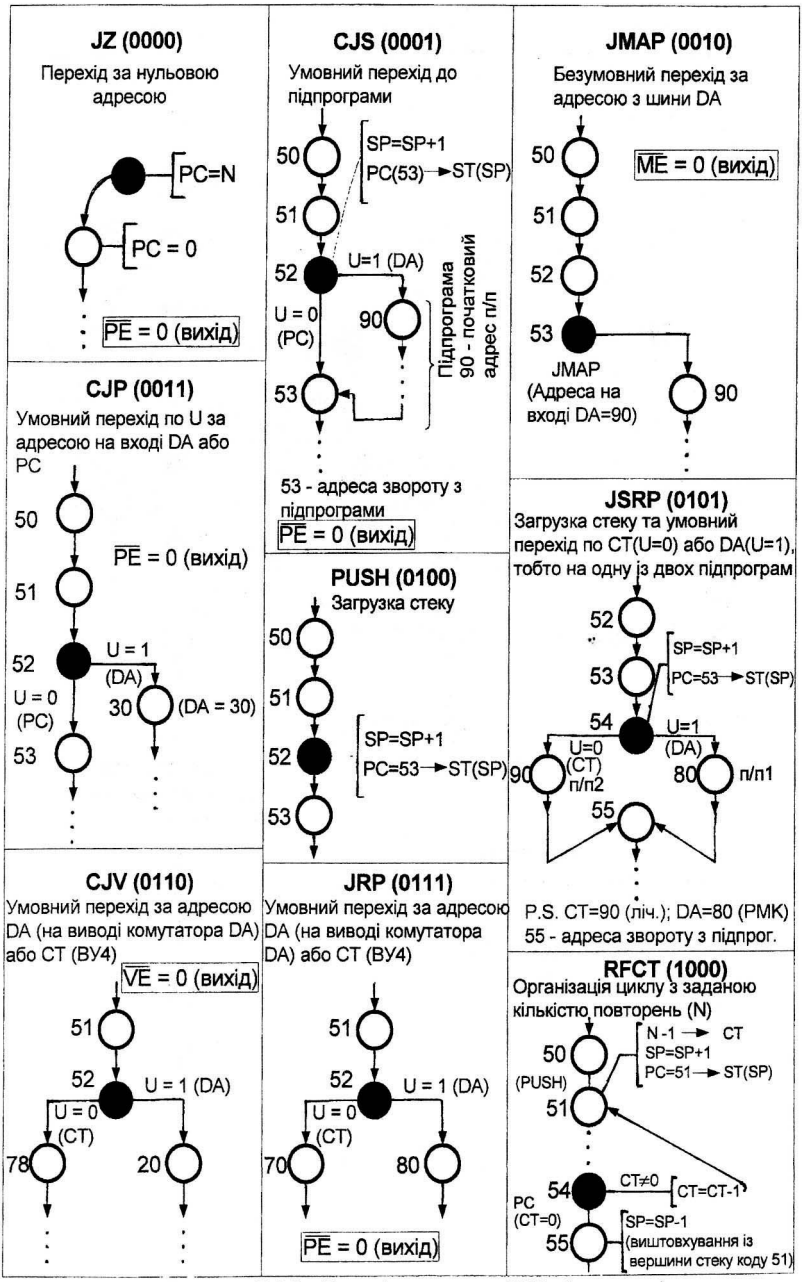



Рисунок Д 2.3. Перелік команд ВУ4 ( початок )

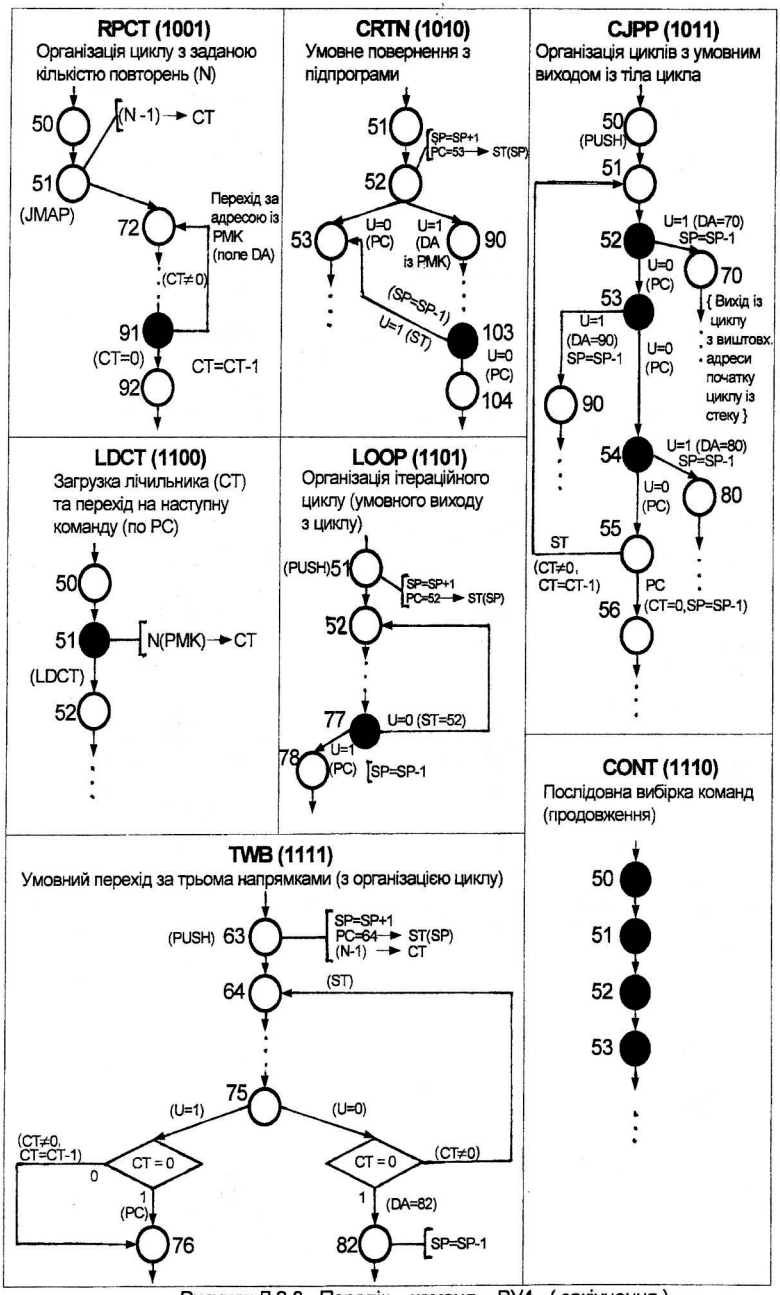



Рисунок Д 2.3. Перелік команд ВУ4 ( закінчення )

## Д2.2 МІКРОПРОЦЕСОРНА СЕКЦІЯ КЕРУВАННЯ АДРЕСОЮ МІКРОКОМАНДИ К1804ВУ1

Мікросхема К1804ВУ1 (надалі ВУ1) має чотири розряди з можливістю нарощування розрядності, кратної чотирьом [1, 2]. Основна функція ВУ1 - формувати адресу МК під впливом зовнішніх керуючих сигналів. Функціональна схема ВУ1 наведена на рис. Д2.4, на якій прийняті наступні позначення сигналів:

**DA (3 – 0)** - код зовнішньої шини адреси. Використовується як одне з чотирьох джерел адреси наступної мікрокоманди.

**R (3 – 0)** - зовнішні сигнали адреси на входах регістра адреси РА (використовуються при завантаженні адреси від зовнішнього джерела в РА).

$\overline{\textbf{RE}}$ - сигнал дозволу запису в РА (рівень активності сигналу низький).

$\overline{\textbf{ZA}}$ - сигнал установи нульової адреси мікрокоманди (звичайно використовується для переходу на початок мікропрограми).

**OR (3 – 0)** - сигнали маскування розрядів адреси (використовуються для установи в одиничний стан окремих розрядів адреси МК).

**S1, S0** - сигнали керування мультиплексором (МХ). Використовуються для формування джерела адреси наступної мікрокоманди.

$\overline{\textbf{OE}}$ - вхідний сигнал дозволу вибору адреси на вихідну шину **Y (3 – 0)**. Використовується для відмикання буферного підсилювача з трьома станами (БПУ).

$\overline{\textbf{FE}}$ - сигнал дозволу роботи зі стеком.

**PUP** - сигнал керування стеком - задає режим завантаження (**PUP** = 1) або витягу (**PUP** = 0).

**C0** - сигнал вхідного переносу в інкрементор **INC**. Якщо сигнал **C0** =1 → **АМК : = АМК + 1**; якщо **C0** = 0, адреса **АМК** (вихід **INC**) не змінюється.

**C4** - сигнал вихідного переносу з **INC** лічильника адреси мікрокоманд. Сигнали **C0** та **C4** використовуються для нарощування кількості секцій ВУ1 (одна секція ВУ1 може адресувати тільки 16 мікрокоманд).

**CLK** - тактовий сигнал синхронізації блоків ВУ1.

**Y (3 – 0)** - вихідна шина адреси мікрокоманд.

У залежності від значень зовнішніх керуючих сигналів **S1** та **S0** (табл. Д2.2) мультиплексор МХ вибирає джерело адреси наступної мікрокоманди (зовнішня шина **DA**, осередок стека **ST**, регістр адреси **РА** або лічильник адреси мікрокоманд **РАМК**).

Керування мультиплексором адреси МХ. Таблиця Д2.2

| S1 | S0 | Вихід МХ (А) |
|---|---|---|
| 0 | 0 | РАМК |
| 0 | 1 | РА |
| 1 | 0 | ST |
| 1 | 1 | DA |

На виході МХ включені чотири двовходові схеми АБО, які дозволяють модифікувати обрану адресу **А** за допомогою сигналів маскування **OR ( 3 –0 )**. Далі за схемою включені чотири двовходові кон'юнктора, які керуються сигналом $\overline{\textbf{ZA}}$. При значенні сигналу $\overline{\textbf{ZA}}$ = 0 на виході **АМК** формується код нульової адреси незалежно від обраного джерела та результату операції маскування.

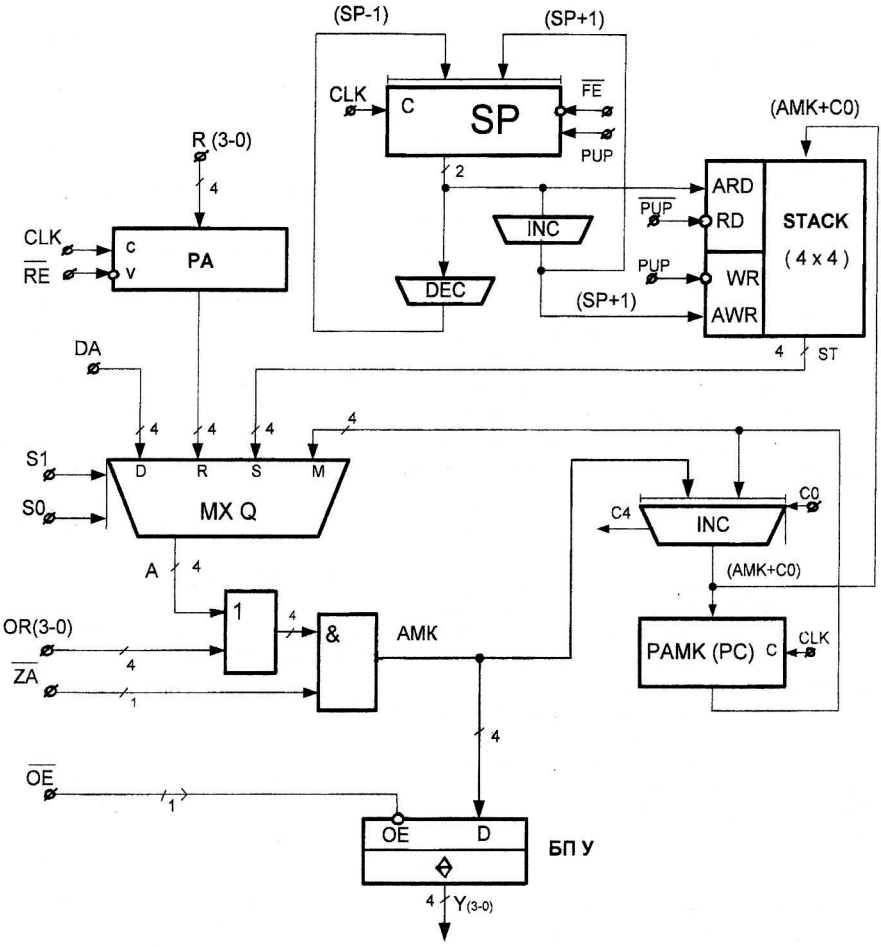



Рисунок Д2.4. Функціональна схема СКАМ К1804ВУ1

Сформований код адреси **АМК** може бути виданий на вихід **Y (3 – 0)** буферного підсилювача з трьома станами БПУ, керований сигналом $\overline{\mathbf{OE}}$. Спільна дія сигналів **OR (i)**, $\overline{\mathbf{ZA}}$ та $\overline{\mathbf{OE}}$ відображена в табл. Д2.3.

Керування виводами Y (3 – 0) ВУ1. Таблиця Д2.3

| OR (i) | $\overline{\mathbf{ZA}}$ | $\overline{\mathbf{OE}}$ | Вихід Y (i), i = 3, 2, 1, 0 |
|---|---|---|---|
| X | X | 1 | HZ |
| X | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | * |

**Примітка.** * - джерело обирається сигналами **S1** та **S0** (табл. Д2.2);
X = 0 або 1; HZ - третій стан.

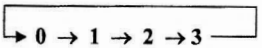
Запис інформації в РА відбувається по позитивному фронту (0 → 1) тактового сигналу **CLK** при низькому рівні сигналу $\overline{\mathbf{RE}}$ ($\overline{\mathbf{RE}}$ = 0). Регістр **РА** затримує адресу **R** на один такт синхросигнала. Аналогічним чином по позитивному фронту сигнала **CLK** відбувається запис інформації з виходу інкрементора **INC** у регістр РАМК лічильника адреси мікрокоманд.

Стек містить нагромаджувач (**STACK**), покажчик стека (**SP**) та схеми запису / читання інформації. Нагромаджувач має чотири комірки по чотири розряди кожна. Керування роботою стека здійснюється за допомогою сигналів $\overline{\mathbf{FE}}$ та **PUP** (табл. Д2.4).

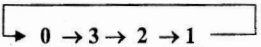
Керування роботою стека ВУ1. Таблиця Д2.4

| $\overline{\mathbf{FE}}$ | PUP | Операція зі стеком |
|---|---|---|
| 1 | X | Стек відключений |
| 0 | 1 | PUSH (завантаження в стек) |
| 0 | 0 | POP (витяг зі стека) |

Стек виконаний як стек кільцевого типу. При виконані операції завантаження **PUSH** по позитивному фронту тактового сигналу **CLK** відбувається зміна покажчика стека **SP** за схемою:




У випадку виконання операції витяга **POP** по позитивному фронту сигналу **CLK** покажчик **SP** змінюється за наступним правилом:




При значенні сигналу $\overline{\mathbf{FE}}$ = 1 (значення сигналу **PUP** при цьому байдуже) відбувається операція читання інформації з вершини стека без зміни вмісту **SP**.

У формуванні адреси наступної МК беруть участь сигнали на виводах ВУ1: **C0**, **OR (3 –0)**, **S1**, **S0**, **PUP**, $\overline{\mathbf{FE}}$ та $\overline{\mathbf{ZA}}$. Послідовність формування адреси залежить від сигналів **C0** та **OR (3 – 0)**. Ці сигнали можуть бути сформовані з РМК. Сигнали на виводах **S1**, **S0**, **PUP** та $\overline{\mathbf{FE}}$ визначають адресу **A** на виході **MX** та режим роботи стека.

Вони можуть бути отримані на виводах кодового перетворювача (**КП**), який являє собою КЛС (рис. Д2.5). Вхідними сигналами **КП** є код інструкції ВУ1 **MI (3 – 0)** та код умови **U**, яка перевіряється. У табл. Д2.5, як приклад, наведені декілька можливих кодів інструкцій з коментарями.

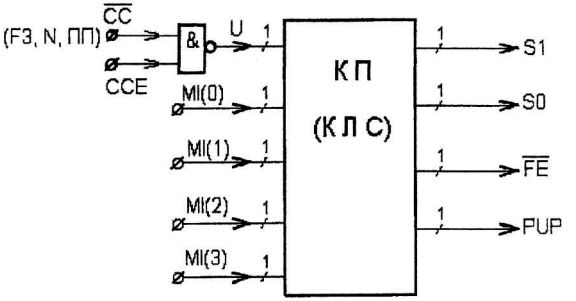



Рисунок Д.2.5 Організація кодового перетоворювача сигналів S1, S0, $\overline{FE}$ та PUP

Таблиця істиності роботи КП. Таблиця Д2.5

| U | MI (3-0) | S1 | S0 | $\overline{FE}$ | PUP | Примітки |
|---|---|---|---|---|---|---|
| * | 0 0 1 0 | 1 | 1 | 1 | * | **A := DA** |
| 0 | 0 0 1 1 | 0 | 0 | 1 | * | **A := PAMK , при U = 0,** |
| 1 | 0 0 1 1 | 1 | 1 | 1 | * | **A := DA , при U = 1** |
| * | 0 1 0 0 | 0 | 0 | 0 | 1 | **A := PAMK, PUSH** |
| 0 | 0 0 0 1 | 0 | 0 | 1 | * | A := PAMK , при U = 0, |
| 1 | 0 0 0 1 | 1 | 1 | 0 | 1 | A := DA , PUSH , при U = 1 |

Для формування адрес, розрядність яких більше чотирьох, необхідно об'єднати декілька мікросхем ВУ1. Так, наприклад, блок із трьох мікросхем дозволяє адресувати МПП обсягом в 4 К мікрокоманд (це робить одна мікросхема ВУ4). При нарощувані необхідно з'єднати не тільки шини керування (**S1, S0, $\overline{FE}$, PUP, CLK, $\overline{RE}$, $\overline{OE}$**), але і лінії **C0 → C4** сусідніх мікросхем ВУ1 (рис. Д2.6). При розрахунку параметрів тактового сигналу **CLK** варто забезпечити достатній час для поширення переносу від входу **ВхП** до виходу **ВихП**.

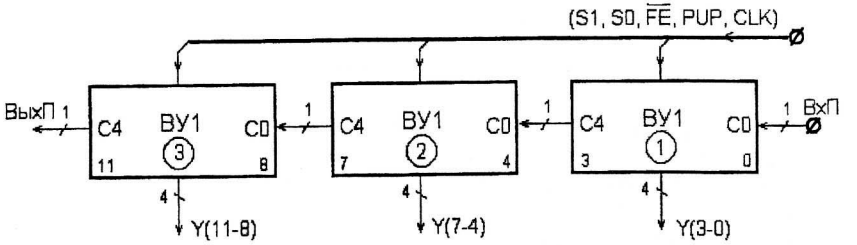



Рисунок Д2.6. Об'єднання секцій ВУ1 для формування 12-ої адреси

## Д2.3 Схема керування наступною адресою К1804ВУ3

Схема керування наступною адресою К1804ВУ3 призначена для перетворення коду інструкції мікрокоманди MI (3 - 0) та входу умови U, яка перевіряється, у набір сигналів для керування блоками на основі ВІС К1804ВУ1 [1 - 4]. Основою мікросхеми ВУ3 є комбінаційний перетворювач (КП), який має п'ять вхідних виводів і вісім вихідних виводів (рис. Д2.7). Перетворювач виконаний у вигляді ПЗП на 32 комірки по вісім розрядів кожна. На виводах КП підключені вісім буферних підсилювачів з трьома станами, керування якими здійснюється за допомогою зовнішнього керуючого сигналу $\overline{OE}$ (може надходити з РМК). При низькому рівні сигналу $\overline{OE}$ буферні підсилювачі знаходяться у відкритому стані (пропускають на свої виводи сигнали з КП).

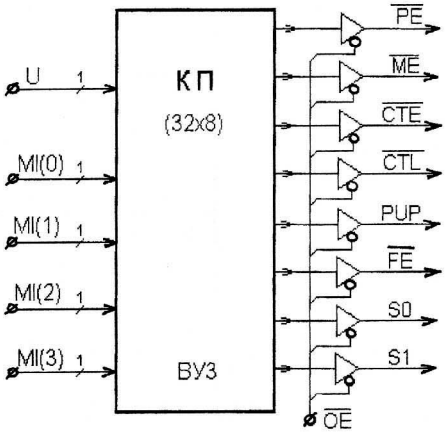



Рисунок Д.2.7 Структурна схема ВІС К1804ВУ3

Сигнали з виходу КП S1, S0, PUP та $\overline{FE}$ можуть бути безпосередньо заведені на відповідні входи мікросхем К1804ВУ1. Сигнали $\overline{CTE}$ та $\overline{CTL}$ можуть бути використані для керування зовнішнім лічильником подій. Сигнали $\overline{PE}$ і $\overline{ME}$ за аналогією з однойменними сигналами мікросхеми ВУ4 можуть бути використані для керування буферними підсилювачами при подачі на вхід DA двох адрес від зовнішніх джерел.

Схема ВУ3 дозволяє реалізувати 16 інструкцій керування послідовністю мікрокоманд. Набір цих інструкцій ідентичний набору операцій, реалізованих мікросхемою К1804ВУ4. Відрізняється тільки операція з кодом MI = 15. У мікросхемі ВУ3 ця операція позначається як JP і є безумовним переходом за адресою, обираною з регістра мікрокоманд.

У табл. Д2.6 мнемонічні позначення інструкцій наведені разом з таблицею істиності комбінаційного перетворювача ВУ3.

Інструкції мікросхеми ВУЗ та їхні коди. Таблиця Д2.6

| Мнемонічні позначення | MI (3) | MI (2) | MI (1) | MI (0) | U | S1 | S0 | $\overline{FE}$ | PUP | $\overline{CTL}$ | $\overline{CTE}$ | $\overline{ME}$ | $\overline{PE}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **JZ** | 0 | 0 | 0 | 0 | * | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| **CJS** | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| | | | | | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| **JMAP** | 0 | 0 | 1 | 0 | * | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| **CJP** | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| | | | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| **PUSH** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| | | | | | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| **JSRP** | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| | | | | | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| **CJV** | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| | | | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **JRP** | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | | | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| **RFCT** | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| | | | | | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| **RPCT** | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| | | | | | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| **CRTN** | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| | | | | | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| **CJPP** | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| | | | | | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| **LDCT** | 1 | 1 | 0 | 0 | * | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| **LOOP** | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| | | | | | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| **CONT** | 1 | 1 | 1 | 0 | * | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| **JP** | 1 | 1 | 1 | 1 | * | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

Умовні графічні позначення ВІС блоків мікропрограмного керування наведено на рис. Д2.8.

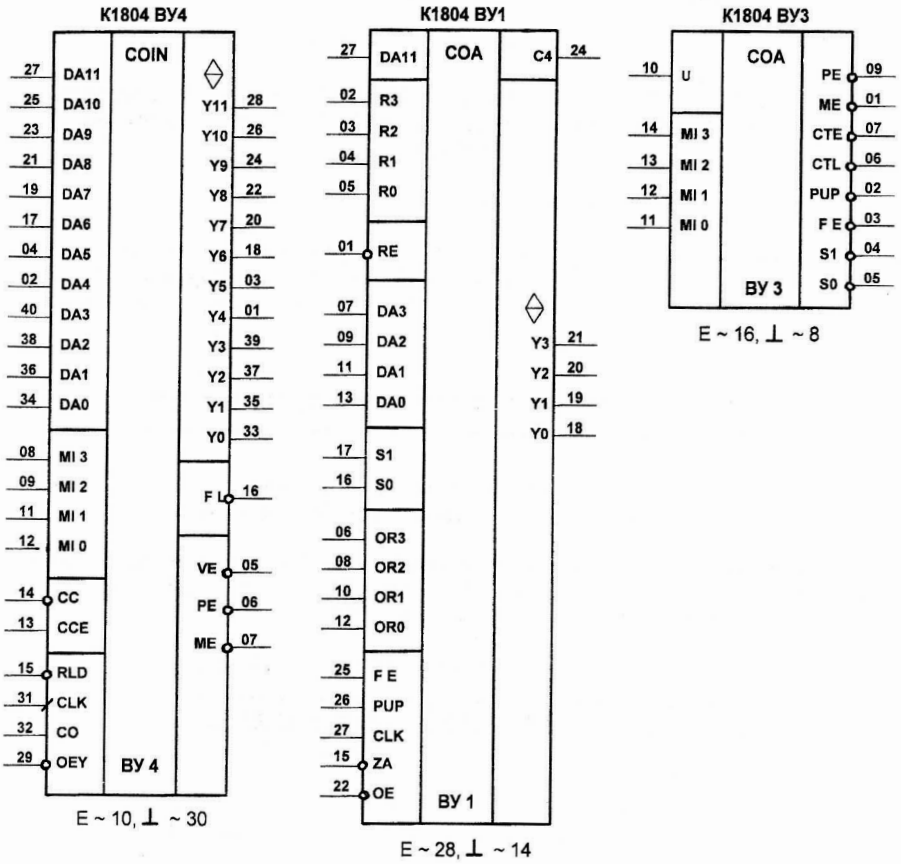



Рисунок Д2.8. Умовне графічне позначення ВІС блоків керування

## Д3. ЗАВДАННЯ ТА МЕТОДИЧНІ ВКАЗІВКИ
**для самостійної роботи студентів денної та заочної форм навчання**

### Д3.1 Контрольна робота № 1

## МІКРОПРОЦЕСОРНИЙ ОБЧИСЛЮВАЛЬНИЙ ПРИСТРІЙ НА ОСНОВІ МПК К1804 ДЛЯ ВИКОНАННЯ АРИФМЕТИЧНИХ ОПЕРАЦІЙ З РУХОМОЮ КОМОЮ

### Індивідуальні завдання до роботи № 1

Розробити функціональну схему (ФС) мікропроцесорного обчислювального пристрою (МОП), граф – схему алгоритма (ГСА) та таблицю кодування (ТК) для виконання заданої операції над числами у форматі з рухомою комою, який наведено на рис. Д3.1.

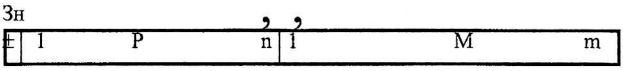



Рисунок Д3.1. Формат чисел із рухомою комою.
( P - порядок числа; M - мантиса числа; Зн - знак числа).

Припустити, що синхронізація усіх блоків МОП відбувається тактовим сигналом CLK. Початкові дані вибрати з табл. Д3.1 - табл. Д3.10. В структурі МОП також передбачити регістри мікрокоманд (РМК) та станів (РП).

Алгоритм МОП. Таблиця Д3.1

| (N)m4 | Назва операції з рухомою комою |
|---|---|
| 0 | Складання |
| 1 | Множення |
| 2 | Віднімання |
| 3 | Ділення |

Довжина порядку числа ( n ). Таблиця Д3.2

| (N)m5 | "n" |
|---|---|
| 0 | 5 |
| 1 | 6 |
| 2 | 7 |
| 3 | 8 |
| 4 | 9 |

Довжина мантиси числа ( m ). Таблиця Д3.3

| (N)m4 | Значення "m" |
|---|---|
| 0 | 12 |
| 1 | 14 |
| 2 | 16 |
| 3 | 18 |

Основа характеристики ( S ) числа з рухомою комою. Таблиця Д3.4

| (N)m3 | "S" |
|---|---|
| 0 | 2 |
| 1 | 4 |
| 2 | 8 |

Зображення порядку числа з рухомою комою. Таблиця Д3.5

| (N)m3 | Зображення порядку числа |
|---|---|
| 0 | Додатковий код (ДК) |
| 1 | Код з позитивним нулем (ПН) |
| 2 | Код з негативним нулем (НН) |

Зображення мантиси числа з рухомою комою. Таблиця Д3.6

| (N)m2 | Зображення мантиси числа |
|---|---|
| 0 | Прямий код (ПК) |
| 1 | Додатковий код (ДК) |

Метод прискорення та алгоритм множення мантис. Таблиця Д3.7

| (N)m4 | Метод та алгоритм множення |
|---|---|
| 0 | Лемана, "А" |
| 1 | Карцева, "В" |
| 2 | Мак – Сорлі, "Г" |
| 3 | Бута, "Б" |

Алгоритм ділення мантис. Таблиця Д3.8

| (N)m2 | Алгоритм ділення |
|---|---|
| 0 | "а" |
| 1 | "б" |

Мікропроцесорні елементі МОП. Таблиця Д3.9

| (N)m4 | Типи процесорних елементів |
|---|---|
| 0 | К1804ВС1, К1804ВУ1 |
| 1 | К1804ВС2, К1804ВУ4 |
| 2 | К1804ВС1, К1804ВУ4 |
| 3 | К1804ВС2, К1804ВУ1 |

Операнди для контрольного прикладу в звіті. Таблиця Д3.10

| (N)m3 | Операнди A і B | |
|---|---|---|
| 0 | A = - 20.75 | B = + 0.25 |
| 1 | A = - 17. 25 | B = + 3.5 |
| 2 | A = - 35.75 | B = - 27.25 |

**Зміст звіту до роботи № 1**

1. Цифрова діаграма реалізації алгоритму заданої операції у двійковій системі (по крокам) з перевіркою результату в 10-вій системі числення.
2. Функціональна схема МОП з коротким описом її роботи.
3. ГСА з коментарями.
4. Таблиця кодування МПП з коментарями.
5. Формули для розрахунку тривалості такту роботи МОП.

## МЕТОДИЧНІ ВКАЗІВКИ ДО РОБОТИ № 1

Для підготовки звіту до контрольної роботи доцільно ознайомитися з методичними вказівками до виконання лабораторних робіт за №10 та №11.

### Д3.2. Контрольна робота №2

## АЛГОРИТМИ ЦЕНТРАЛЬНОГО ПРОЦЕСОРНОГО ПРИЛАДУ (ЦПП) ДЛЯ ОБРОБКИ КОМАНД ДОВІЛЬНОГО ФОРМАТУ

**Індивідуальні завдання до роботи № 2**

Розробити функціональну схему ЦОМ та ЦПП і граф – схему алгоритма вибірки з оперативної пам'яті (ОП) машини та виконання команди заданного формату (рис. Д3.2). Початкові дані для виконання роботи наведені в табл. Д3.11 та табл. Д3.12. У форматі заданої команди скласти програму з коментарями, яка обчислює наступну функцію:

$$Y = \begin{cases} a_i \cdot x + b_{i+1}, & x \geq 0; \\ a_i + c, & x < 0, \end{cases}$$

Припустимо, що параметр **i** змінюється в діапазоні від 5 до (N + 5), де N - номер варіанта завдання. Приняти, що змінна **x** розташована у 100 комірки ОП.

Обов'язкові операції ЦОМ. Таблиця Д3.11

| (N)m3 | Мнемоничне позначення операцій |
|---|---|
| 0 | АО, ПО, УПN |
| 1 | АО, БП, УПZ |
| 2 | АО, ІПО, УПV |

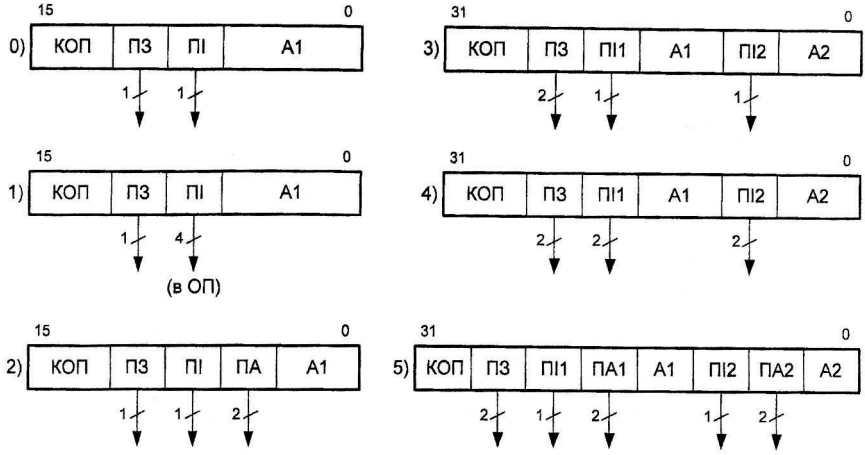



Рисунок Д3.2. Формати команд ЦОМ ( варіант вибрати як (N) mб ).

**Примітки.** КОП - код операції; ПЗ - признак запису результата операції в ОП; ПІ - признак індексації; ПА - признак способа адресації операнда; А1 або А2 - адреса або операнд; АО - арифметична операція; БП - безумовний перехід; УПN, УПZ, УПV - умовний перехід відповідно по ознаку знаку результату (N), нульового результату (Z) та переповнення (OVR) відповідно. Кодування полів команд відображено в таблицях Д3.13 – Д3.17.

Ємність модуля оперативної пам'яті (ОП). Таблиця Д3.12

| (N)m5 | Ємність (в кілобайтах) та довжина (в бітах) комірки ОП |
|---|---|
| 0 | 64К x 8 |
| 1 | 8К x 16 |
| 2 | 16К x 32 |
| 3 | 8К x 8 |
| 4 | 16К x 64 |

### Зміст звіту до роботи № 2

1. Функціональна схема ЦОМ та ЦПП. В коментарях вказати призначення усіх керуючих, синхронізуючих та інформаційних сигналів.
2. Граф – схема алгоритма виконання операцій в ЦПП ЦОМ із краткими коментарями.
3. Програма на рівні команд ЦОМ для реалізації функції Y з використанням ЦОМ, яка була розроблена при виконанні контрольної роботи.

## МЕТОДИЧНІ ВКАЗІВКИ ДО РОБОТИ № 2

Для підготовки звіту до лабораторної роботи доцільно ознайомитися з методичними вказівками до виконання лабораторної роботи за №12.

Кодування поля признака адресації (ПА) команди. Таблиця Д3.13

| ПА | Спосіб адресації операнда |
|---|---|
| 0 0 | **Пряма** (адреса операнда задається в полі А1 (або А2) команди) |
| 0 1 | **Безпосередьня** (в полі А1 (або А2) команди знаходиться операнд) |
| 1 0 | **Побічна** (в полі А1 (або А2) команди знаходиться адрес адреса операнда) |
| 1 1 | Не використовується |

Кодування поля признака запису результату (ПЗ) при виконанні операції в одноадресних командах. Таблиця Д3.14

| ПЗ | Виконуєма дія |
|---|---|
| 0 | ОП ( А1 ) → РР ( АЛП ) |
| 1 | РР ( АЛП ) → ОП ( А1 ) |

**Примітка.** РР - регістр результату АЛП.

Кодування поля признака запису результату (ПЗ) при виконанні операцій в двоадресних командах. Таблиця Д3.15

| ПЗ | Виконуєма дія |
|---|---|
| 0 0 | ( А1 ) * ( А2 ) → ( А2 ) |
| 0 1 | ( А1 ) * ( А2 ) → РР |
| 1 0 | РР * ( А1 ) → ( А2 ) |
| 1 1 | РР * (А1 ) → РР |

**Примітка.** * - операція АЛП.

Кодування поля признака індексації (ПІ) в форматах команд 0 та 3. Таблиця Д3.16

| ПІ | Формування виконавчої адреси Авик |
|---|---|
| 1 | Авик = А1 + ІР |
| 0 | Авик = А1 |

Кодування поля признака індексації (ПІ) в форматах команд 2 та 4. Таблиця Д3.17

| ПІ | Формування виконавчої адреси Авик |
|---|---|
| 0 0 | Авик = А1 |
| 0 1 | Авик = А1 + ІР1 |
| 1 0 | Авик = А1 + ІР2 |
| 1 1 | Авик = А1 + ІР3 |

**Примітка.** ІР1, ІР2, ІР3 - індексні регістри ЦОП. В команді формату 1 індексні регістри розташовані в перших 15 комірок оперативній пам'яті (ОП). Поле ПІ команди в цьому разі завдає адрес індексного регістра в ОП.

# СПИСОК ЛІТЕРАТУРИ

## 1. Література з мікропроцосорів серії К1804


1.Проектирование цифровых систем на комплектах микропрограммируемых БИС / С.С. Булгаков, В. М. Мещеряков, В.В. Новоселов, Л.А. Шумилов; Под ред. В. Г. Колесникова – М.: Радио и связь, 1984. – 240 с.

2.Жабин В. И. Однокристальные и микропрограммируемые ЭВМ. – К.: Диалектика, 1995. – 116 с.

3.Мик Дж., Брик Дж. Проектирование микропроцессорных устройств с разрядно – модульной организацией – М.: Мир, 1984. – т.1, т.2.

4.Хвощ С.Т., Варлинский Н.Н., Попов Е.А. Микропроцессоры и микроЭВМ в системах автоматического управления – Л.: Машиностроение, 1988. – 640 с.

5.Справочник по устройствам цифровой обработки информации / Н.А. Виноградов, В. Н. Яковлев, В.В. Воскресенский и др. – К.: Техніка, 1988. - 415 с.

6.Микросхемы серии К1804. – Северодонецк: РП НПО "Импульс".


## 2. Література з опису і застосуванню мови VHDL


7.Семенец В. В., Хаханова И. В., Хаханов В. И. Проектирование цифровых систем с использованием языка VHDL: Учебное пособие – Харьков: ХНУРЭ, 2003. – 492 с.

8.Суворова Е. А., Шейнин Ю. Е. Проектирование цифровых систем на VHDL – СПб.: БХВ – Петербург, 2003. – 576 с.

9.Сергиенко А. М. VHDL для проектирования вычислительных устройств –К.: ЧП "Корнейчук', ООО "ТИД ДС", 2003. – 208 с.

10.Поляков А. К. Языки VHDL и VERILOG в проектировании цифровой аппаратуры – М.: СОЛОН –Пресс, 2003. – 320 с.

11.Кондратенко Ю П., Сидоренко С. А., Підопригора Д. М. Поведінковий синтез цифрових пристроїв у середовищі Active – HDL: Навчальний посібник / За ред. Ю. П. Кондратенка - Миколаєв: Вид – во МФ НА УКМА, 2002. – 116 с.

12.Бибило П. Н. Синтез логических схем с использованием языка VHDL – М.: - СОЛОН – Р, 2002. – 384 с.

13.Армстронг Дж. Р. Моделирование цифровых систем на языке VHDL: Пер. с англ. – М.: Мир, 1992. – 175 с.

14.Яицков А. С. VHDL - язык описания аппаратных средств: Учебное пособие / Под ред. акад. В. С. Бурцева, акад. Б. С. Митина – М.: Изд – во МАТИ – РГТУ "ЛАТМЭС", 1998. – 119 с.

15.VHDL'92. Новые свойства языка описания аппаратуры VHDL / Ж. М. Берже. А. Фонкуа, С. Мажиро, Ж. Руйар: Пер. с англ. – М.: Радио и связь, 1995. – 256 с.

15.Поляков А. К. Моделирование ЭВМ на языке VHDL / Под ред. Ю. А. Татарникова – М.: Изд – во МЭИ, 1994. – 107 с.

17.Соловьев В. В. Проектирование цифровых систем на основе программируемых логических интегральных схем – М.: Горячая линия – Телеком, 2001. - 636 с.

18.Бибило П. Н. Основы языка VHDL – Минск: Нн – т техн. кибер. НАН Беларуси, 1999. – 202 с.

19.VHDL для моделирования, синтеза и формальной верификации аппаратуры / Пер. с англ. - М.: Радио и связь, 1995. – 360 с.

20.Стешенко В. Б. Практика автоматизированного проектирования радиоэлектронных устройств – М.: "Нолидж", 2002. – 768 с.

21.Грушницкий Р. И., Мурсаев А. Х., Угрюмов Е. П. Проектирование систем на микросхемах программируемой логики – СПб.: БХВ – Петербург, 2002. – 608 с.

## ЗМІСТ

# МЕТОДИЧНІ ВКАЗІВКИ

**ДО ЛАБОРАТОРНОГО ПРАКТИКУМУ**
**з курсу "Архітектура комп'ютерів" для підготовки спеціалістів та магистрів на основі напрямку "Комп'ютерна інженерія"**

**УКЛАДАЧІ:** Володимир Васильович Лапко;
Юрій Володимирович Губарь