2. МІКРОПРОЦЕСОРИ НА ОСНОВІ СЕКЦІОНОВАНИХ ІНТЕГРАЛЬНИХ СХЕМ

2.1. Загальні відомості

Секціоновані мікропроцесорні серії інтегральних схем (ІС) призначені для розробки МПС з будь-якою системою команд і довільною розрядністю.

Найпоширенішим набором таких ІС є серія К1804, яка складається з наступних ІС:

К1804ГГ1 – генератор тактових сигналів
К1804ВА1, К1804ВА2, К1804ВАЗ – чотирирозрядні приємопередавачі;
К1804ВЖ1 – шістнадцятирозрядна схема виправлення помилок в ЗП;
К1804ВН1 – блок пріоритетного переривання на 8 запитів;
К1804ВР1 – схема прискореного переносу;
К1804ВР2 – блок управління станами та зсувами;
К1804ВР3 – розширювач блока пріоритетного переривання;
К1804ВС1, К1804ВС2 – чотирирозрядні процесорні елементи;
К1804ВУ1, К1804ВУ2 – чотирирозрядні генератори адрес мікрокоманд;
К1804ВУ3 – схема управління генератором адрес мікрокоманд;
К1804ВУ4 – дванадцятирозрядний генератор адрес мікрокоманд (формувач адреси наступної мікрокоманди);
К1804ВУ5 – чотирирозрядний генератор адрес мікрокоманд;
К1804ИР1 – чотирирозрядний регістр;
К1804ИР2, К1804ИР3 – восьмирозрядні регістри.

Розглянемо основні ІС серії К1804 та приведемо особливості проектування мікропроцесорних систем на їх основі.

2.2. Процесорний елемент

ІС К1804ВС1 виконана за технологією ТТЛШ і є чотирирозрядним універсальним процесорним елементом (ПЕ), призначеним для

побудови блоків обробки даних. Структурна схема ПЕ зображена на рис. 2.1.

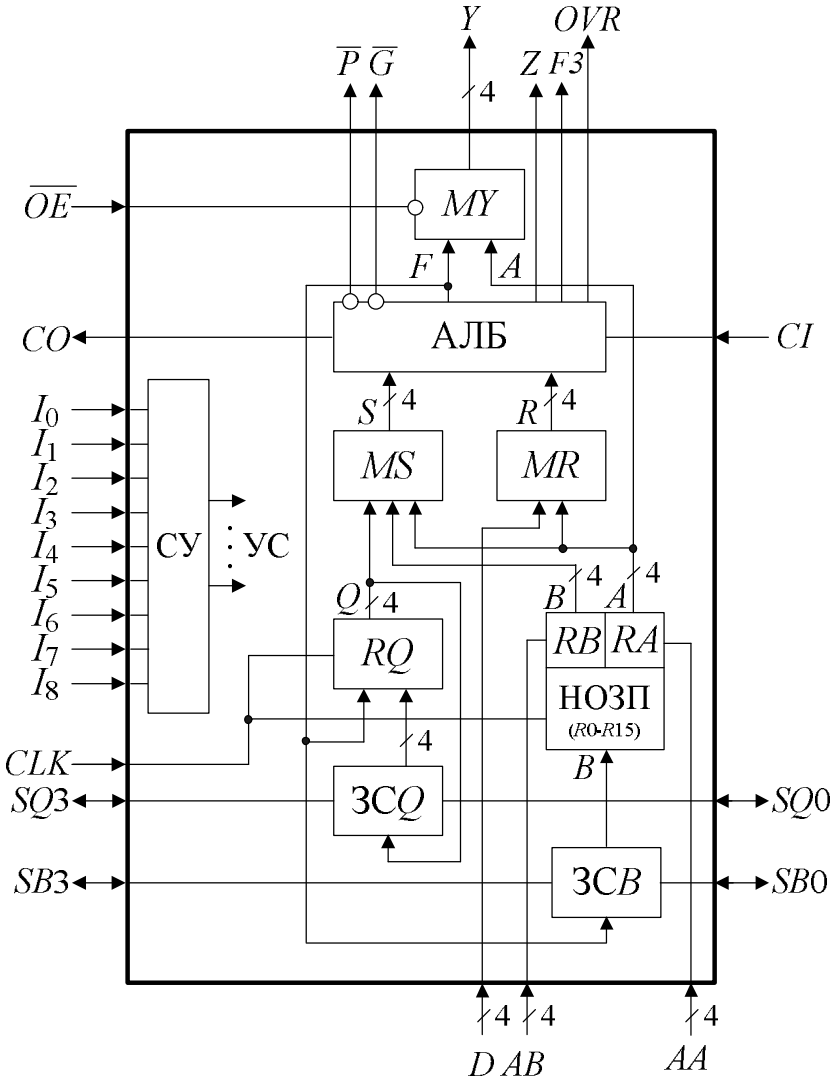


Рис. 2.1. Структурна схема процесорного елементу ІС К1804ВС1

Процесорний елемент К1804ВС1 складається з наступних функціональних частин:

АЛБ – арифметико-логічний блок;
НОЗП – надоперативний запам'ятовуючий пристрій, що складається з шістнадцяти регістрів $R15 - R0$;
RA, *RB* – регістри тимчасового зберігання адрес операндів за каналом *A* і *B*;
ЗС*Q*, ЗС*B* – зсувачі;
MS, MR, *MY* – мультиплексори вибору операндів *S і R* та даних *Y*
СУ – схема управління.

Умовне позначення ПЕ зображене на рис 2.2, призначення виводів мікросхеми описане в табл. 2.1.

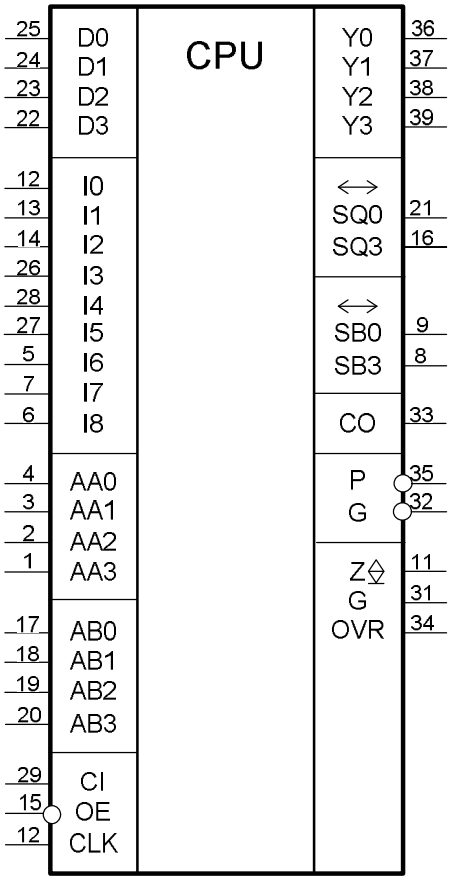


Рис 2.2. Умовне графічне позначення мікросхеми ПЕ К1804ВС1

Молодший розряд ПЕ має номер 0, а старший – номер 3.

Виходи *Y* мають три стани (0, 1 і високоомний). Встановлення в високоомний стан цих виходів здійснюється подачею на вхід *OE* одиничного потенціалу $\overline{OE} = 1$.

Виводи *SQ*3, *SQ*0, *SB*3 і *SB*0 є двоспрямованими і мають також три стани. При виконанні на зсувачах ЗС*B* і ЗС*Q* зсуву вправо виводи *SB*3 і *SQ*3 є входами, а виводи *SB*0 і *SQ*0 – виходами. При зсуві інформації вліво функції виводів змінюються на протилежні. Якщо зсув на зсувачі не виконується, то відповідні йому виводи знаходяться в високоомному стані. Вихід *Z* виконаний за схемою з відкритим колектором. Решта виходів *IC* мають два стани (0 та 1).

Таблиця 2.1. Функціональне призначення виводів мікросхеми ПЕ К1804ВС1

Номери виводів	Позначення	Функціональне призначення
1 – 4	*AA*3 – *AA*0	Входи адрес регістрів НОЗП за каналом *A*
12 – 14, 26, 28, 27, 5 – 7	*I*8 – *I*0	Входи мікрокоманди
8 (9)	*SB*3 (*SB*0)	Вхід (вихід) старшого (молодшого) розряду ЗС*B*
11	*Z*	Вихід ознаки нульового результату в АЛБ
15	*CLK*	Вхід синхросигналів
16 (21)	*SQ*3 (*SQ*0)	Вхід (вихід) старшого (молодшого) розряду ЗС*Q*.
17-20	*AB*0 – *AB*3	Входи адреси регістрів НОЗП за каналом *B*
22 – 25	*D*3 – *D*0	Входи даних
29	*CI*	Вхід переносу в АЛБ
31	*F*3	Вихід старшого розряду АЛБ
32,35	*G*,*P*	Виходи прискореного переносу
33	*CO*	Вихід переносу з АЛБ
34	*OVR*	Вихід ознаки переповнювання результату
36 – 39	*Y*0 – *Y*3	Виходи даних
40	*OE*	Вхід дозволу видачі даних
33	*CO*	Вихід переносу з АЛБ

В ПЕ використовується двоадресний НОЗП, який забезпечує видачу інформації за двома незалежними каналами *A* і *B* (рис. 2.1.). Інформація на виходах *A* і *B* визначається відповідно адресами *AA* і *AB*. Якщо на входах *AA* і *AB* встановлені однакові адреси регістрів, то на виходи *A* і *B* видається вміст одного й того самого регістру. Адреси регістрів дорівнюють двійковим еквівалентам їх номерів. Наприклад, регістр *R*5 має адресу 0101 і таке інше. Запис інформації в НОЗП відбувається за адресою *B* – тільки за одним каналом.

Для управління ПЕ в структурі мікрокоманди відведено 18 розрядів (рис. 2.3).

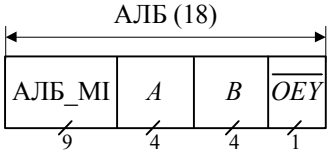


Рис. 2.3. Поля мікрокоманди, які використовуються для управління ПЕ

Характер перетворення інформації в кожнім такті роботи ПЕ визначається інформаційним словом (мікроінструкцією МІ), що розміщується в полі АЛБ_МІ і надходить на входи $I8-I0$ мікросхеми К1804ВС1. Формат інформаційного слова для управління ПЕ наведений на рис. 2.4.

Рис. 2.4. Формат інформаційного слова АЛБ_МІ мікрокоманди

Поле ($I_8 I_7 I_6$) управляє мультиплексором *MY*, зсувачами ЗС*Q* і ЗС*B*, а також вибором приймача результату (рис.2.1). Кодування розрядів даного поля наведене у табл. 2.2.

Результат *F* мікрооперації, що виконувалась в АЛБ, може бути записаний в НОЗП за каналом *B* ($F \to B$) без змін, а також у модифікованому вигляді – із зсувом на один розряд вліво ($2F \to B$) або

вправо ($F/2 \to B$). За встановлення значення $I_8 I_7 I_6 = 000$ результат записується в регістр *RQ* ($F \to Q$). Результат не фіксується в регістрах ПЕ за встановлення значення $I_8 I_7 I_6 = 001$.

Таблиця 2.2. Кодування поля управління мультиплексором

Розряди мікрокоманди			Мікрооперації		*Y*
I_8	I_7	I_6	НОЗП	*RQ*	
0	0	0	–	$F \to Q$	*F*
0	0	1	–	–	*F*
0	1	0	$F \to B$	–	*A*
0	1	1	$F \to B$	–	*F*
1	0	0	$F/2 \to B$	$Q/2 \to Q$	*F*
1	0	1	$F/2 \to B$	–	*F*
1	1	0	$2F \to B$	$2Q \to Q$	*F*
1	1	1	$2F \to B$	–	*F*

Під час встановлення на вході сигналу дозволу видачі даних *OE* ($\overline{OE} = 1$) та за $I_8 I_7 I_6 = 010$ мультиплексор *MY* видає на вихідні шини *Y* інформацію з виходів каналу *A* НОЗП (з регістру *RA*), а в інших випадках – з виходів *F* АЛБ.

Одночасно із записом результату в НОЗП в регістрі *RQ* може бути виконаний зсув інформації вліво ($I_8 I_7 I_6 = 110$) або вправо ($I_8 I_7 I_6 = 100$).

Розряди $I_5 I_4 I_3$ поля АЛБ_МІ МК (рис.2.4) визначають мікрооперацію в АЛБ відповідно до табл. 2.3, де *R* і *S* виходи мультиплексорів *MR* і *MS*.

Мікрооперації підсумовування і віднімання виконуються у доповнювальному коді з урахуванням вхідного переносу *CI*. Логічні мікрооперації – диз'юнкція (АБО), кон'юнкція (І) і сума за модулем два (ВИКЛЮЧНЕ АБО) є порозрядними. Вибір джерел операндів здійснюється за допомогою мультиплексорів *MS* і *MR* та управляється розрядами $I_2 I_1 I_0$ поля АЛБ_МІ мікрокоманди (табл. 2.4).

Таблиця 2.3. Кодування поля $I_5\ I_4\ I_3$

Розряди мікрокоманди			Мікрооперація в АЛБ
I_5	I_4	I_3	
0	0	0	$R+S+CI$
0	0	1	$S-R-1+CI$
0	1	0	$S-R-1+CI$
0	1	1	$R \vee S$
1	0	0	$R \& S$
1	0	1	$\overline{R} \& S$
1	1	0	$R \oplus S$
1	1	1	$\overline{R \oplus S}$

Таблиця 2.4. Кодування поля $I_2\ I_1\ I_0$

Розряди мікрокоманди			Джерела операндів	
I_2	I_1	I_0	R	S
0	0	0	A	Q
0	0	1	A	B
0	1	0	0	Q
0	1	1	0	B
1	0	0	0	A
1	0	1	D	A
1	1	0	D	Q
1	1	1	D	0

Такт роботи ПЕ полягає в наступному.

За додатним перепадом *CLK* на управляючі входи ПЕ подається слово мікрокоманди (рис. 2.1). В регістри *RA* і *RB* з відповідних регістрів НОЗП, які визначаються адресами на входах *AA* і *AB*, записуються дані. Відповідно до значень розрядів $I_8 - I_0$ мікрокоманди схема управління (СУ) виробляє необхідні управляючі сигнали (УС),

в результаті цього обираються джерела операндів. Далі відбувається перетворення операндів в АЛБ і видається результат на вихід *F*. Якщо мультиплексор *MY* відкритий ($\overline{OE} = 0$), результат видається на шину *Y*. Залежно від результату, одержуваного в АЛБ, на відповідних виходах ПЕ формуються ознаки *Z*, *F*3, *OVR*.

Одиничний сигнал на виході *Z* відповідає нульовому значенню результату. Ознакою арифметичного переповнювання в АЛБ є одиничний сигнал на виході *OVR*. Вивід *F*3 є виходом старшого розряду АЛБ і використовується для аналізу знаку результату.

За від'ємним перепадом синхросигналу $CLK = 0$ результат фіксується у вибраному приймачеві результату – в регістрах НОЗП або в регістрі *RQ*. Для забезпечення правильної роботи ПЕ необхідно, щоб сигнали на управляючих входах ПЕ не змінювали свого значення за $CLK = 0$.

Часові параметри ІС К1804ВС1 наведені в табл. 2.5.

Таблиця 2.5. Часові параметри ІС К1804ВС1

Параметри ІС К1804ВС1	Значення, нс	Параметри ІС К1804ВС1	Значення, нс
Тривалість сигналу		від *I* до *У*, *CO*, *F*3	50
$CLK = O$ (*min*)	30	від *I* до *G*, *P*	42
Час попереднього встановлення відносно додатного перепаду *CLK* сигналів на входах (*min*):		від *I* до *Z*	65
		від *I* до *OVR*	59
		від *I* до *SB*3, *SB*0	70
		від *D* до *У*	39
RA, *RB*	93	від *B* до *G*, *D*	31
D, *I*	70	від *D* до *Z*	55
CI	55	від *D* до *CO*, *F*3	41
*SQ*3, *SQ*0, *SB*3, *SB*0	20	від *D* до *OVR*	45
Час затримки		від *D* до *SB*3, *SB*0	53
від *RA*, *RB* до у, *F*3	75	від *CI* до *У*	27
від *RA*, *RB* до *G*, *P*	59	від *C1* до *Z*	46
від *RA*, *RB* до *Z*	85	від *CI* до *CO*	20
від *RA*, *RB* до *CO*	70	від *CI* до *F*3	24
від *RA*, *RB* до *OYR*	76	від *CI* до *OVR*	26
від *RA*, *RB* до *SB*3, *SB*0	90	від *CI* до *SB*3, *SB*0	45

Приклад 2.1. Розробити мікропрограму для процесорного елементу, що реалізує заданий мікроалгоритм (рис. 2.5).

Вихідні дані: $X1 = -3$, $X2 = -9$, $X3 = 11$.

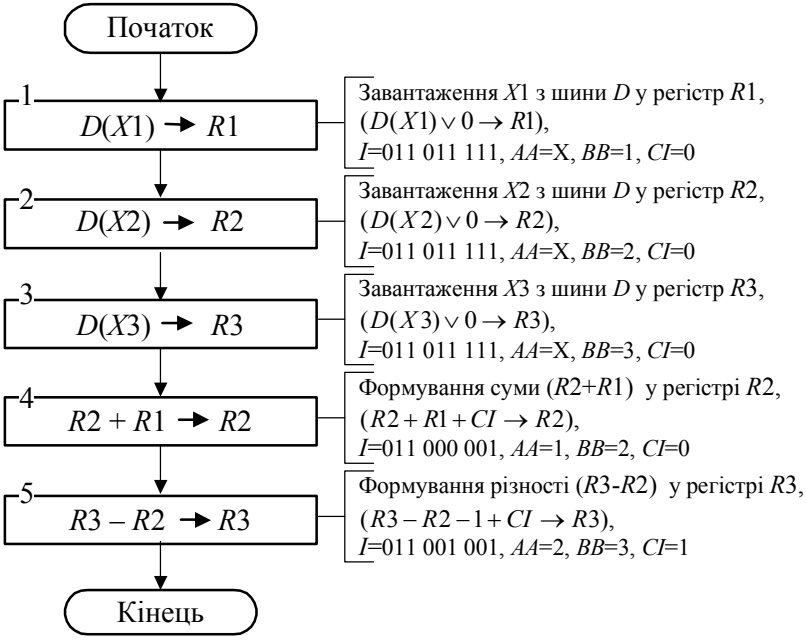


Рис. 2.5. Вихідний мікроалгоритм

Вважатимемо, що для подання операндів використовується доповнювальний код. В коментарях до кожної операторної вершини (рис. 2.5) наведені значення управляючих сигналів АЛБ_МІ, *AA*, *BB*, *CI*, що забезпечують виконання відповідної мікрооперації.

Послідовність мікрокоманд, що реалізують заданий мікроалоритм, відображена у табл. 2.6. Цифрова діаграма стану регістрів показана у табл. 2.7.

Розглянемо мікрооперацію завантаження $D(X1) \to R1$, що виконується в першому такті. Зазначимо, що дані в регістри НОЗП можуть бути завантажені лише, як результат виконання мікрооперації в АЛБ. Для завантаження даних з шини даних виконаємо мікрооперацію логічного додавання. Джерелами даних, що надходять на

входи *S* і *R* АЛБ (рис. 2.1), є машинний нуль і дані на вхідній шині *D* ($D(X1) \vee 0 \to R1$) Відповідно до табл. 2.4. визначаємо розряди $I_2 I_1 I_0$ поля АЛБ_МС мікрокоманди $I_2 I_1 I_0 = 111$. Мікрооперація логічне І, що здійснюється у цьому такті, визначається розрядами $I_5 I_4 I_3 = 011$ поля АЛБ_МС мікрокоманди (табл. 2.3). Результат, отриманий на виході *Y* АЛБ записується в регістр НОЗП за адресою, виставленою на шині *AB*. За умовою завдання результат має бути записаний у регістр *R*1, тому у полі *B* мікрокоманди (рис. 2.9) розміщується адреса відповідного регістру у шістнадцятирічному поданні *B* = 1. При цьому кодуємо розряди $I_8 I_7 I_6$ поля АЛБ_МС мікрокоманди (табл. 2.2), отримуємо $I_8 I_7 I_6 = 011$. Мікрооперація додавання, що здійснюється у четвертому такті, визначається розрядами $I_5 I_4 I_3 = 000$ поля АЛБ_МІ мікрокоманди (табл. 2.3). при цьому встановлюється *C1*=0, що відповідає розрядам $I_{12} I_{11} = 00$ поля СУСЗ_МІ мікрокоманди (табл. 2.19). У полях *A* і *B* мікрокоманди (рис. 2.3) розміщуються номери регістрів НОЗП у шістнадцятирічному поданні, що є джерелами операндів. Результат, отриманий на виході *Y* АЛБ записується в регістр НОЗП за адресою, виставленою на шині *AB*, тому у полі *B* мікрокоманди (рис. 2.3) розміщується номер регістра-приймача результату (*A* = 1, *B* = 2). Мікрооперація віднімання, що здійснюється у п'ятому такті, визначається розрядами $I_5 I_4 I_3 = 001$ поля АЛБ_МІ мікрокоманди (табл. 2.3). при цьому встановлюється *C1*=1, що відповідає розрядам $I_{12} I_{11} = 01$ поля СУСЗ_МІ мікрокоманди (табл. 2.19).

Наведемо мікропрограму у мнемонічних кодах мікроасемблеру:

```
        /-- Завантаження даних в регістри НОЗП---
        accept r1: 0fffdh /--(–3)ДК → R1
        accept r2: 0fff7h /--(–9)ДК → R1
        accept r3: 0000bh /--(11)ДК → R1
0003    /--Область програми---------------------
0004    {add r2,r2,r1,z;}   /--Обчислення суми
        {sub r3,r2,r3,nz;} /--Обчислення різності
        {}
```

Таблиця 2.6. Кодова карта

№ такту	Адреса [16]	Константа БОД		БОД													
				ВС1 (АЛБ)				ВР2 (СУСЗ)									
		D [16]	$\overline{OED}$	АЛБ_МІ [2]	А [16]	В [16]	$\overline{OEY}$	СУСЗ_МІ [2]	$\overline{EC}$	$\overline{EZ}$	$\overline{EN}$	$\overline{EV}$	$\overline{CEN}$	$\overline{CEM}$	$\overline{OECT}$	$\overline{SE}$
1	0000	*FFFD*	0	011 011 111	*	1	0	00 00000 000000	0	0	0	0	1	1	1	1
2	0001	*FFF*7	0	011 011 111	*	2	0	00 00000 000000	0	0	0	0	1	1	1	1
3	0002	000*B*	0	011 011 111	*	3	0	00 00000 000000	0	0	0	0	1	1	1	1
4	0003	****	1	011 000 001	1	2	0	00 00000 000000	0	0	0	0	1	1	1	1
5	0004	****	1	011 001 001	2	3	0	01 00000 000000	0	0	0	0	1	1	1	1

Таблиця 2.7. Цифрова діаграма стану регістрів

№ так-ту	Вихідний стан				Інформація у приймачі результату, [16]	Мікрооперація
	Шина ***D***, [16]	***R*1**, [16]	***R*2**, [16]	***R*3**, [16]		
1	*FFFD*	****	****	****	$R1 := FFFD$	$R1 := D(FFFD) \vee 0$
2	*FFF*7	*FFFD*	****	****	$R2 := FFF7$	$R2 := D(FFF7) \vee 0$
3	000*b*	*FFFD*	*FFF*7	****	$R3 := 000B$	$R3 := D(000B) \vee 0$
4	****	*FFFD*	*FFF*7	000*B*	$R2 := FFF4$	$R2 := R2 + R1 + 0(CI)$
5	****	*FFFD*	*FFF*4	000*B*	$R3 := 0017$	$R3 := R2 - R3 - 1 + 1(CI)$

.

2.3. Схема управління станами та зсувами

2.3.1. Загальна структура та принцип функціонування

Схема управління станами та зсувами (СУСЗ) призначена для спільної роботи з ПЕ К1804ВС1, разом з якими складає блок обробки даних (БОД). Схема управління станами та зсувами реалізована на ІС К1804ВР2, структурна схема якої наведена на рис. 2.6. Загальна структура БОД та поєднання виводів мікросхем К1804ВР2 та наведені на рис. 2.7.

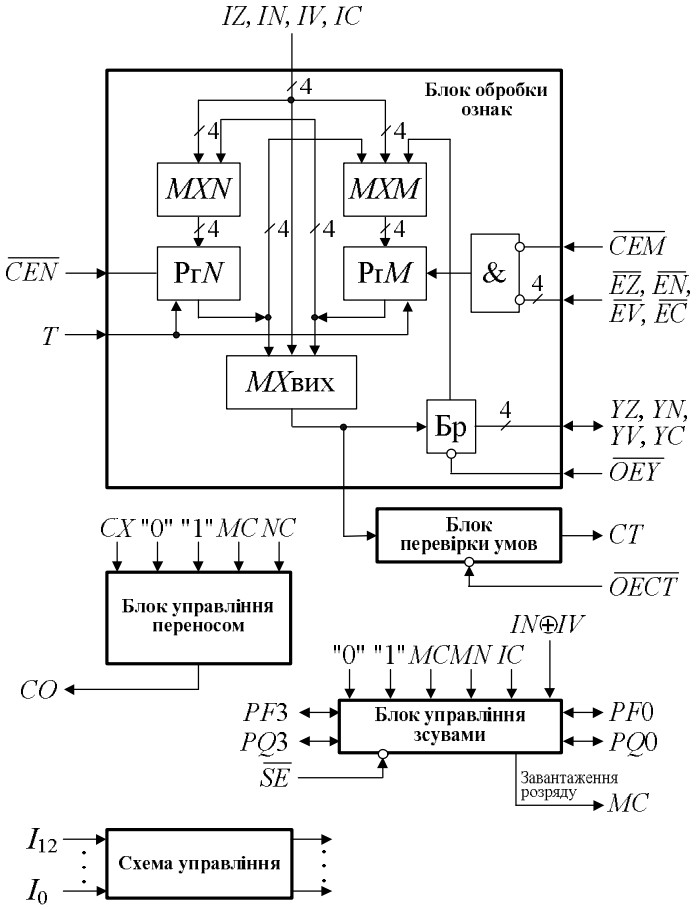


Рис. 2.6. Структурна схема СУСЗ ІС К1804ВР2

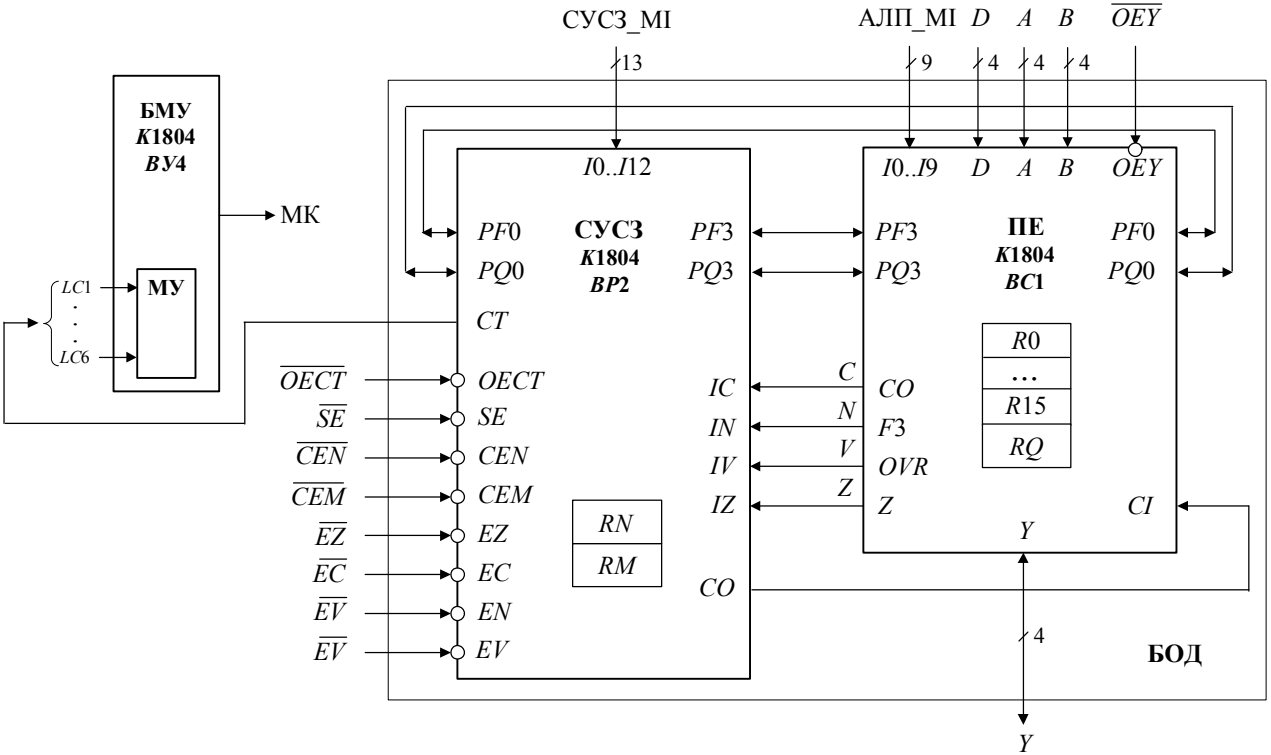


Рис. 2.7. Блок обробки даних

Для управління БОД у структурі мікрокоманди відведено 43 розряди. На рис. 2.8 зображені відповідні поля МК.

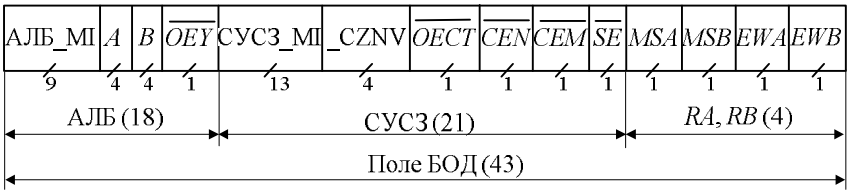


Рис. 2.8. Поля мікрокоманди, які використовуються для управління блоком обробки даних

Умовне графічне позначення ІС К1804ВР2 та нумерація виводів приведені на рис. 2.9, а їх функціональне призначення – в табл. 2.7.

Таблиця 2.7. Функціональне призначення виводів ІС К1804ВР2

Позначення	**Назва**	**Призначення**
$I12 - I0$	Інформаційні входи МК	Визначають операцію, що виконується СУСЗ
IC, IN, IV, IZ	Входи ознак стану	Використовуються для передачі ознак стану АЛБ (переносу, знаку, переповнювання, рівність нулю) в СУСЗ
$\overline{CEM}$	Вхід дозволу запису ознак в РгM	При значенні 0 на вході $\overline{CEM}$ ($\overline{CEM} = 0$) запис в РгM дозволений, інакше ($\overline{CEM} = 1$) – заборонений
$\overline{CEN}$	Вхід дозволу запису ознак в PrN	При значенні 0 на вході $\overline{CEN}$ ($\overline{CEN} = 0$) запис в PrN дозволений, інакше ($\overline{CEN} = 1$) – заборонений
$\overline{EC}$, $\overline{EN}$, $\overline{EV}$, $\overline{EZ}$	Входи дозволу запису ознак	При значенні 1 на входах $\overline{EC}$, $\overline{EN}$, $\overline{EV}$, $\overline{EZ}$ запис у відповідні розряди РгM заборонений, а при значенні 0 – дозволений, якщо одночасно встановлений сигнал дозволу запису $\overline{CEM}$ =0

YC, *YN*, *YV*, *YZ*	Двоспрямована шина даних	Застосовується як вхід даних під час запису в регістр Рг*M* і як вихід даних під час виводу інформації з регістрів *PrN*, Рг*M* або з входів *IC*, *IN*, *IV*, *IZ* СУСЗ
$\overline{OEY}$	Вхід управляючого сигналу дозволу виводу інформації	За значення 0 на вході $\overline{OEY}$ ($\overline{OEY}=0$) дозволений вивід інформації зі СУСЗ через шину *Y*. При $\overline{OEY}=1$ шина *Y* знаходиться в стані високого опору – вивід даних на шину *Y* заборонений
$\overline{OECT}$	Вхід управляючого сигналу дозволу формування коду умови	За значення 0 на вході $\overline{OECT}$ ($\overline{OECT}=0$) дозволений вивід коду умови на вихід *CT*. Якщо $\overline{OECT}=1$, формування коду умови на виході *CT* заборонене
CT	Вихід умови	Результат перевірки умови
*PF*0, *PF*3, *PQ*0, *PQ*3	Двоспрямовані виводи зсуву	Призначені для організації зсувів в МПС шляхом поєднання їх із відповідними виводами зсувів
$\overline{SE}$	Вхід дозволу зсуву	За значення 0 на вході $\overline{SE}$ ($\overline{SE}=0$) виконання зсувів дозволено, інакше ($\overline{SE}=1$) – заборонено, під час цього виводи зсуву СУСЗ знаходяться в стані високого опору
CO	Вихід формування переносу	Сформований на виході *CO* сигнал використовується в якості вхідного переносу *CI* АЛБ
CX	Вхід формування переносу	Використовується як одне з джерел при формуванні сигналу вихідного переносу *CO*
CLK	Тактовий вхід	Запис інформації в регістр стану відбувається за додатним перепадом тактового сигналу *CLK*. Решта схем є комбінаційними – їх функціонування не залежать від значення сигналу *CLK*

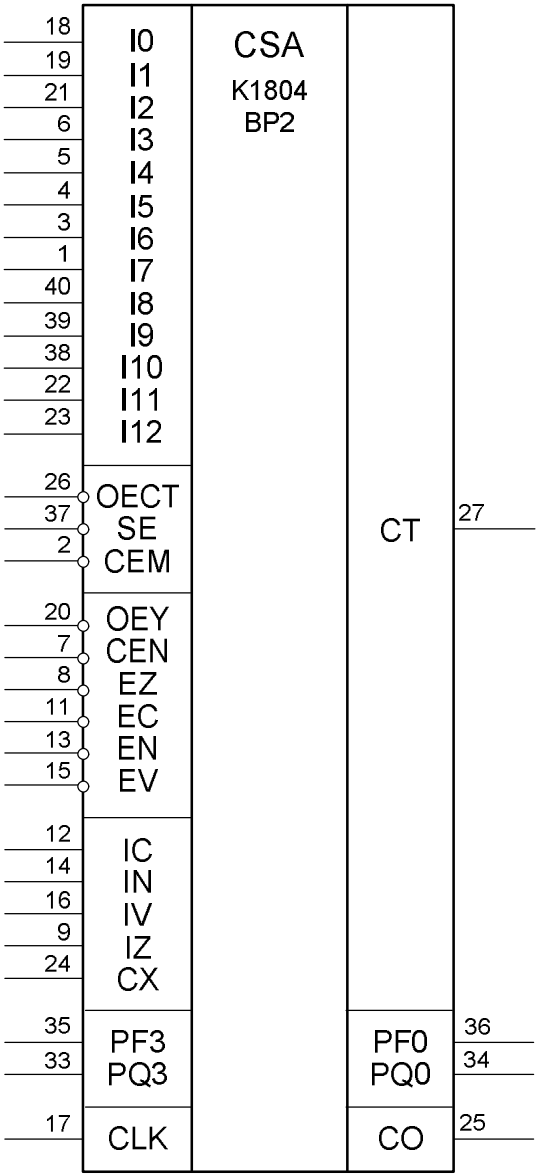


Рис. 2.9. Умовне графічне позначення ІС К1804ВР2

Функціональні блоки, що входять до складу СУСЗ (рис. 2.6) реалізують наступні функції:

– виконання операцій із вмістом регістрів станів (операцій з ознаками);

– формування сигналу вхідного переносу *CI* для ПЕ і СУП;

– організація арифметичних, логічних і циклічних зсувів слів різної довжини на підставі ознак *Z*, *C*, *N*, *V* результату виконання мікрооперації у АЛБ – реалізує 32 типи зсувів;

– видача одного з шістнадцяти сигналів логічної умови на вихід *CT*, який надходить на БМУ і використовується для організації розгалуження у мікропрограмах.

Розглянемо детально склад та призначення функціональних блоків СУСЗ.

2.3.2. Схема управління

Схема управління (рис.2.6) під дією сигналів мікрокоманди $I_{12} - I_0$ формує внутрішні сигнали, що управляють функціональними блоками СУСЗ.

Для управління СУСЗ в структурі мікрокоманди (рис.2.8) відведено 21 розряд, 13 з яких займає поле інформаційного слова СУСЗ_ІС, що задає операцію виконувану в СУСЗ у поточному такті (рис. 2.10).

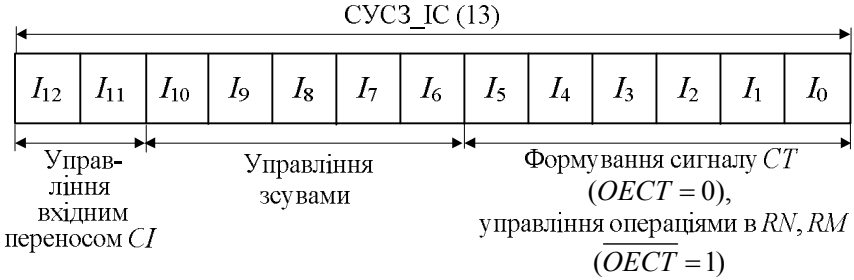


Рис. 2.10. Структура інформаційного поля СУСЗ_ІС мікрокоманди

2.3.3. Блок обробки ознак

Блок обробки ознак (рис. 2.6) складається з двох чотирирозрядних регістрів стану (Рг*N*, Рг*M*), двох вхідних мультиплексорів – *MXN*, *MXM* і одного вихідного мультиплексора – $MX_{\text{ВИХ}}$. Даний блок призначений для зберігання і модифікації ознак, які формуються в ПЕ. Під час виконання мікрооперацій у АЛБ формуються ознаки – переносу (*C*), знаку результату (*N*), переповнювання (*OVR*) і

рівність результату нулю (Z), які з виходів C, N, V, Z ПЕ поступають на входи IC, IN, IV, IZ ознак стану СУСЗ (рис. 2.1, рис. 2.6).

Ознаки зберігаються в регістрах стану РгN і РгM СУСЗ (рис. 2.11). Запис інформації в регістри РгN і РгM відбувається за додатним перепадом тактового імпульсу T (перехід від 0 до 1) за наявності сигналу дозволу запису $\overline{CEN} = 0$ *або* $\overline{CEM} = 0$ відповідно.

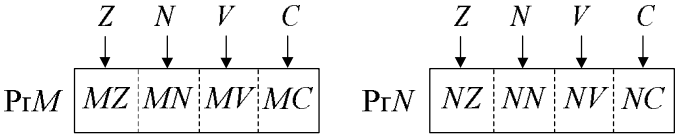


Рис. 2.11. Структура регістрів стану РгN і РгM

2.3.4. Виконання операцій з вмістом регістрів станів

У полі інформаційного слова СУСЗ_ІС якому відповідають розряди $I_5 - I_0$ (рис. 2.10) при встановленні управляючого сигналу $\overline{OECT} = 1$ кодуються операції що можуть бути виконані із вмістом регістрів стану (рис. 2.12).

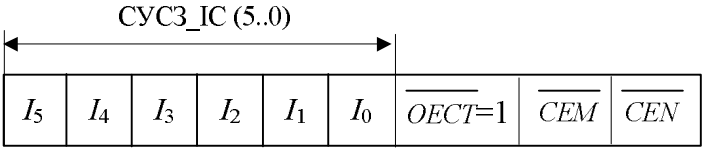


Рис. 2.12. Управління виконанням операцій із вмістом регістрів станів

Операції, що виконуються із вмістом регістрів станів РгN і РгM, можна розділити на наступні групи:

– операції з бітами,
– регістрові операції,
– операції завантаження регістра.

Операції з вмістом регістру стану* Рг*N

В РгN інформація надходить з виходу двоканального мультиплексора MXN (рис. 2.6).

Залежно від сигналів мікрокоманди $I_5 - I_0$ в РгN може бути записане слово стану з входів ознак стану СУСЗ – IC, IN, IV, IZ, або з виходів регістру РгM – MC, MN, MV, MZ. Окрім того, може бути

встановлено в значення 0 або 1, як усе інформаційне слово записане у регістрі Рг*N*, так і кожний окремий розряд цього регістру.

Під час запису інформації у регістр Рг*N* на вході дозволу запису встановлюється значення 0 ($\overline{CEN} = 0$). При $\overline{CEN} = 1$ запис в Рг*N* заборонений.

Операції з бітами відповідають встановленню в значення 0 або 1 одного з розрядів Рг*N* в залежності від сигналів $I_5 - I_0$ закодованих у мікрокоманді (табл. 2.9).

Регістрові операції є операціями з інформаційним словом, записаним в Рг*N*. Управління виконанням операцій здійснюється розрядами I_1 і I_0 мікрокоманди за встановленням нульових значень розрядів мікрокоманди $I_5 - I_2$ (табл. 2.10).

Таблиця 2.9. Операції з бітами регістра стану *PrN* ($\overline{CEN} = 0$)

I_5	I_4	I_3	I_2	I_1	I_0	Операція	Коментар
0	0	1	0	0	1	$1 \to NZ$	Встановлення в 1 ознаки *Z*
0	0	1	0	1	0	$0 \to NC$	Встановлення в 0 ознаки *C*
0	0	1	0	1	1	$1 \to NC$	Встановлення в 1 ознаки *C*
0	0	1	1	0	0	$0 \to NN$	Встановлення в 0 ознаки *N*
0	0	1	1	0	1	$1 \to NN$	Встановлення в 1 ознаки *N*
0	0	1	1	1	0	$0 \to NV$	Встановлення в 0 ознаки *OVR*
0	0	1	1	1	1	$1 \to NV$	Встановлення в 1 ознаки *OVR*

Таблиця 2.10. Регістрові операції з вмістом регістру стану *PrN*, де $i = Z, C, N, V$ ($\overline{CEN} = 0$)

I_5	I_4	I_3	I_2	I_1	I_0	Операція	Коментар
0	0	0	0	0	0	$M_i \to N_i$	Запис вмісту регістру Рг*M* у регістр Рг*N*
0	0	0	0	0	1	$1 \to N_i$	Встановлення в значення 1 всіх розрядів регістру Рг*N*
0	0	0	0	1	0	$M_i \leftrightarrow N_i$	Регістровий обмін між регістрами Рг*M* та Рг*N*
0	0	0	0	1	1	$0 \to N_i$	Встановлення в значення 0 всіх розрядів регістру *PrN*

Операції завантаження регістра відповідають запису ознак стану в Рг*N* з входів ознак стану (*IC*, *IN*, *IV*, *IZ*) (табл. 2.11).

Таблиця 2.11. Операції завантаження регістра стану *PrN* ($\overline{CEN} = 0$)

I_5	I_4	I_3	I_2	I_1	I_0	Операція	Коментар
0 0	0 0	0 0	1 1	1 1	0 1	$IZ \to NZ$ $IC \to NC$ $IN \to NN$ $IV \vee NV \to NV$	Використовується при виконанні ряду арифметичних операцій коли немає необхідності перевіряти ознаку переповнювання після кожної операції, а достатньо встановити, що переповнювання було хоча б під час однієї з операцій
0 1 1 0	1 0 1 0	1 1 1 0	0 0 0 1	0 0 0 0	* * * *	$IZ \to NZ$ $\overline{IC} \to NC$ $IN \to NN$ $IV \to NV$	Завантаження з інверсією ознаки переносу
0 0 0 1 1 1 1 1 1	1 1 1 0 0 0 1 1 1	0 1 1 0 1 1 0 1 1	* 0 1 * 0 1 * 0 1	* 1 * * 1 * * 1 *	* * * * * * * * *	$IZ \to NZ$ $IC \to NC$ $IN \to NN$ $IV \to NV$	Завантаження безпосередньо з входів ознак стану

Операції з вмістом регістру стану* Рг*M

В Рг*M* інформація надходить з виходів мультиплексора *MXM* (рис. 2.7). Залежно від значення сигналів $I_5 - I_0$ в Рг*M* може бути записана інформація з входів ознак стану СУСЗ – *IC*, *IN*, *IV*, *IZ* (ціле слово стану, або порозрядно), з виходів регістру Рг*N* – *NC*, *NN*, *NV*, *NZ* або з двоспрямованої шини *Y* – *YC*, *YN, YV, YZ*. Окрім того в кожен розряд Рг*M* може бути записане значення 0 або 1. Для виконання запису необхідно, щоб на вході дозволу запису в Рг*M* було встановлене значення $\overline{CEM} = 0$. При $\overline{CEM} = 1$ запис в Рг*M* заборонений.

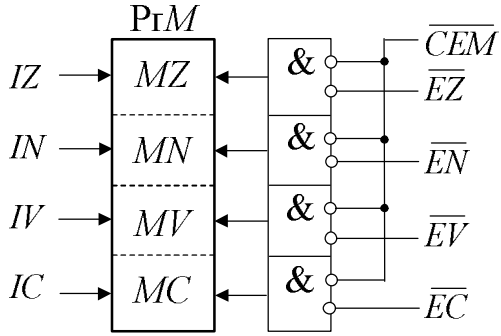


Рис. 2.13. Операції з бітами у регістрі Рг*M*

Операції *запису у біти* регістру Рг*M* ознак результату виконуються з встановленням сигналів дозволу запису $\overline{EZ} = \overline{EN} = \overline{EV} = \overline{EC} = 0$ (рис. 2.13). За встановлення управляючого сигналу $\overline{CEM} = 0$ і наявністю сигналу дозволу запису у біти регістру відбувається запис ознаки у відповідний розряд регістру Рг*M*. Якщо сигнал на вході дозволу запису ознаки $\overline{CEM} = 0$, а значення будь-яких сигналів дозволу запису бітів дорівнює 1, то вміст відповідних розряду регістру Рг*M* не зміниться. Тобто, під час встановлення сигналів $\overline{EZ} \vee \overline{EN} \vee \overline{EV} \vee \overline{EC} = 1$ відбувається заборона запису у відповідний розряд регістру Рг*M*. За значення $\overline{CEM} = 1$ забороняється запис у всі розряди регістру Рг*M*.

Управління виконанням регістрових операції і операції завантаження, що виконуються у регістрі Рг*M*, наведено в табл. 2.12 – 2.13.

Таблиця 2.12. Регістрові операції регістра Рг*M*
($\overline{EZ} = \overline{EN} = \overline{EV} = \overline{EC} = \overline{CEM} = 0$)

I_5	I_4	I_3	I_2	I_1	I_0	Операція	Коментар
0	0	0	0	0	0	$Y_i \to M_i$	Запис інформації з шини *Y* в Рг*M*
0	0	0	0	0	1	$1 \to M_i$	Встановлення в 1 всіх розрядів Рг*M*
0	0	0	0	1	0	$N_i \leftrightarrow M_i$	Регістровий обмін
0	0	0	0	1	1	$0 \to M_i$	Встановлення в 0 всіх розрядів Рг*M*
0	0	0	1	0	1	$\overline{M_i} \to M_i$	Інвертування вмісту регістру Рг*M*

Таблиця 2.13. Операції завантаження регістру Рг*M*
($\overline{EZ} = \overline{EN} = \overline{EV} = \overline{EC} = \overline{CEM} = 0$)

I_5	I_4	I_3	I_2	I_1	I_0	Операція	Коментар
0	0	0	1	0	0	$IZ \to NZ$ $IV \to MV$ $IN \to MN$ $IC \to MC$	Використовується при організації зсувів з використанням ознаки переповнювання, а не переносу
0	0	1	0	0	*	$IZ \to MZ$ $\overline{IC} \to MC$ $IM \to MN$ $IV \to MV$	Завантаження з інверсією ознаки переносу
0	1	1	0	0	*		
1	0	1	0	0	*		
1	1	1	0	0	*		
0	0	0	1	1	*	$IZ \to MZ$ $IC \to MC$ $IN \to MN$ $IV \to MV$	Завантаження безпосередньо з входів ознак стану
0	0	1	0	1	*		
0	0	1	1	*	*		
0	0	1	1	*	*		
0	1	0	*	*	*		
0	1	1	0	1	*		
0	1	1	1	*	*		
1	0	0	*	*	*		
1	0	1	0	1	*		
1	0	1	1	*	*		
1	1	0	*	*	*		
1	1	1	*	*	*		

Приклад 2.2. Розробити мікропрограму для БОД, що реалізує заданий мікроалгоритм (рис. 2.14).

Виконання завдання

У заданому мікроалгоритмі на рис. 2.14 мнемоніка *LOAD* визначає операції завантаження регістрів стану (*RN, RM*) або відповідних бітів цих регістрів (*MC*, *MZ* та інших). У табл. 2.14 наведена кодова карта мікропрограми. У структурі закодованих мікрокоманд умовно показані тільки ті поля, що управляють АЛБ та СУСЗ. В табл. 2.15 наведена діаграма вмісту регістру стану *RM* на протязі виконання мікропрограми.

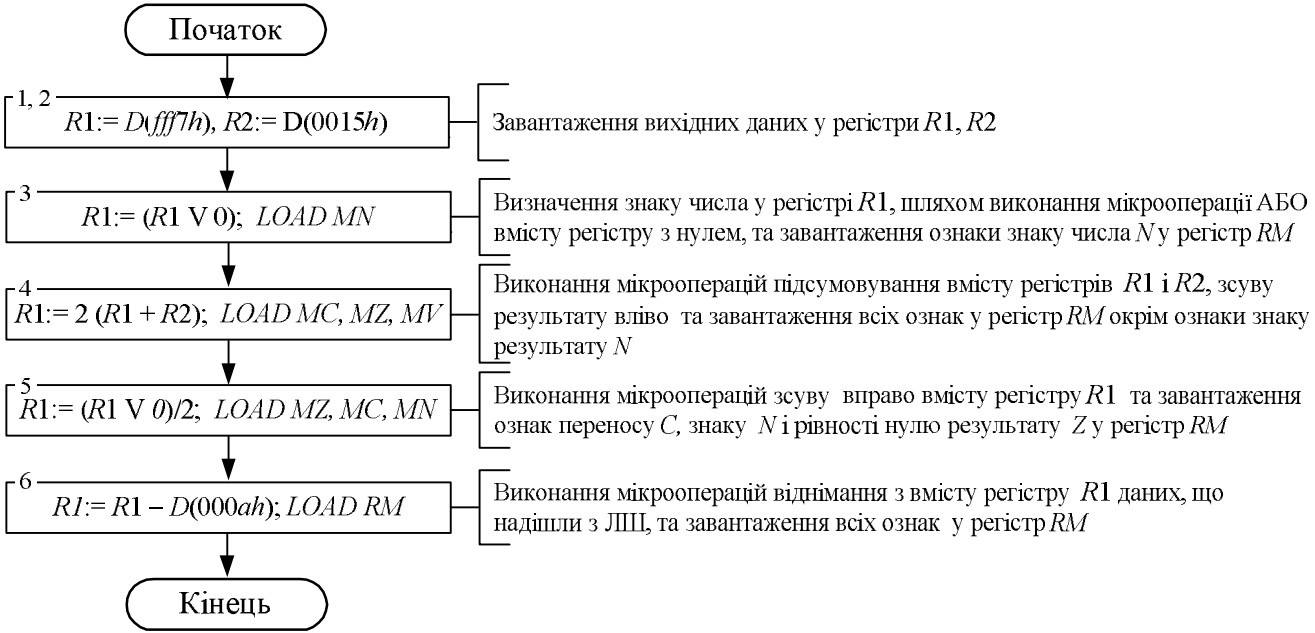


Рис. 2.14. Вихідний мікроалгоритм

Таблиця 2.14. Кодова карта

№ такту	Адреса [16]	Константа БОД		БОД												
				ВС1 (АЛБ)				ВР2 (СУСЗ)								
		D [16]	$\overline{OED}$	АЛБ_МІ [2]	A [16]	B [16]	$\overline{OEY}$	СУСЗ_МІ [2]	$\overline{EC}$	$\overline{EZ}$	$\overline{EN}$	$\overline{EV}$	$\overline{CEN}$	$\overline{CEM}$	$\overline{OECT}$	$\overline{SE}$
1	0001	*FFF7*	0	011 011 101	1	1	1	00 00000 000000	0	0	0	0	1	1	1	1
2	0002	0015	0	011 011 101	2	2	1	00 00000 000000	0	0	0	0	1	1	1	1
3	0003	****	1	011 011 100	1	1	1	00 00000 000110	1	1	0	1	1	0	1	1
4	0004	****	1	111 000 001	2	1	1	00 10000 000110	0	0	1	0	1	0	1	0
5	0005	****	1	101 011 100	1	1	1	00 00010 000110	0	0	0	1	1	0	1	0
6	0006	001*A*	0	011 001 101	1	1	1	01 00000 000110	0	0	0	0	1	0	1	1

Таблиця 2.15. Діаграма вмісту регістру стану *RM*

№ такту	*MC*	*MZ*	*MN*	*MV*
3	*	*	1	*
4	1	0	*	0
5	0	0	*	0
6	0	0	1	0

Результат налагодження мікропрограми у моделюючій програмі *COMPLEX* представлений на рис. 2.15.

COMPLEX

Файл Редактор Выполнить Протокол Помощь Опции Выход

Исходный файл :SUSZ_K.PMK Страница: 1.1

Адр	Конст. БОД БМУ	OED	Б О Д ВС1 MI*	A	B	OEY	ВР2 MI*	C	Z	N	V	OECT	CEN	CEM	SE
000	FFF7	0	011.011.101	1	1	1	00.00000.000000	0	0	0	0	1	1	1	1
001	0015	0	011.011.101	2	2	1	00.00000.000000	0	0	0	0	1	1	1	1
002	0000	1	011.011.100	1	1	1	00.00000.000110	1	1	0	1	1	1	0	1
003	0000	1	111.000.001	2	1	1	00.10000.000110	0	0	1	0	1	1	0	1
004	0000	1	101.011.100	1	1	1	00.00010.000110	0	0	0	1	1	1	0	1
005	001A	0	011.001.101	1	1	1	01.00000.000110	0	0	0	0	1	1	0	1
006	0000	1	001.000.000	0	0	1	00.00000.000000	0	0	0	0	1	1	1	1

Исходный файл :SUSZ_K.PMK Страница: 1.2

Адр	РАА,РАВ MSA	MSB	EWA	EWB	БМУ и БПП ВУ4 MI*	CCE	COM	CI	RLD	MS	ВН1 MI*	EINS	ПАВ EV	ОП,ВУ,РА ВУ I	O	LCK	IA	ОП R	W	РгА EWH	EWL
000	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
001	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
002	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
003	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
004	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
005	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
006	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1

Нажмите любую клавишу для окончания

Исходный файл :SUSZ.ASM Страница: 1

```
accept r1:0fff7h
accept r2:0015h
{or r1,r1,z;load rm,flags;cem_c;cem_z;cem_v;}
{add sll,r1,r1,r2,z;load rm,flags;cem_n;}
{or srl,r1,r1,z;load rm,flags;cem_v;}
{sub r1,r1,001ah,nz;load rm,flags;}
{}
```

Рис. 2.15. Мікропрограма управління станами

Вивід вмісту регістрів* Рг*N*, Рг*M* *на двоспрямовану шину Y

Інформація з виходів Рг*N*, Рг*M* або з входів ознак стану СУСЗ – *IC*, *IN*, *IV*, IZ, через вихідний мультиплексор із трьохстабільними виходами $МХ_{Вих}$ (рис. 2.6) надходить на двоспрямовану шину *Y*. Якщо на входах $I_5 - I_0$ встановлені нульові значення, то шина *Y* є вхідною незалежно від сигналу дозволу виводу інформації ($\overline{OEY}$). В інших випадках шина *Y* є вихідною. Управління виводом інформації через шину *Y* за допомогою сигналів $\overline{OEY}$, I_5 та I_4 надане в табл. 2.16.

Таблиця 2.16. Управління виводом інформації через шину *Y* ($i = Z, C, N, V$)

$\overline{OEY}$	I_5	I_4	Y	Коментар
1	*	*	Z	Стан високого опору
0	0	*	$N_i \to Y_i$	Вивід вмісту Рг*N*
0	1	0	$M_i \to Y_i$	Вивід вмісту Рг*M*
0	1	1	$I_i \to Y_i$	Вивід інформації з входів ознак стану
*	$(I_5..I_0) = 000000$ $Y_i \to$ Рг*M*			При значенні 0 на входах $I_5 - I_0$ шина *Y* є вхідною незалежно від значення сигналу $\overline{OEY}$

2.3.5. Блок перевірки умови

Блок перевірки умови (рис. 2.6) складається з комбінаційної схеми перевірки умови та мультиплексора зі схемою управління полярністю.

Блок перевірки умови призначений для формування вихідного сигналу коду умови *CT*, яка надходить на один з входів мультиплексора умов МУ (рис. 2.7) БМУ і використовується для організації різноманітних переходів у мікропрограмах.

Управління формуванням умови здійснюється розрядами $I_5 - I_0$ поля СУСЗ_IC МК (рис. 2.10) за встановлення сигналу $\overline{OECT} = 0$ (рис. 2.16).

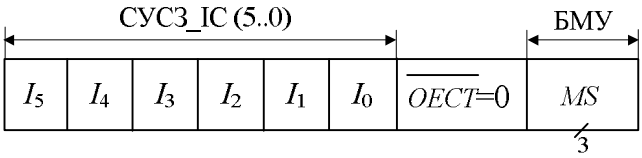


Рис. 2.16. Управління формуванням сигналу *CT*

Блок перевірки умови виконує одну з 16 можливих операцій (табл. 2.17) формування умови, результат якої видається на вихід коду умови *CT*. Вибір операндів для виконання операцій в даному блоці управляється сигналами мікрокоманди I_4, I_5.

Найбільш поширеними операціями, що виконуються блоком перевірки умови є передавання однієї з ознак стану на вихід *CT* (чотири операції). Ще чотири операції є більш складними і застосовуються під час виконання в АЛБ операції віднімання ($A - B$) для забезпечення умов $A = B$, $A \neq B$, $A \geq B$ та інших. (табл. 2.18). При цьому аргументи *A* і *B* мають бути подані в доповнювальному коді або як числа без знаку.

Далі результат однієї з виконаної у блоці операції формування умови через мультиплексор передається на вхід ***схеми управління полярністю***, яка за необхідності інвертує отриманий результат. Таким чином решта вісім операцій є інверсією вже розглянутих операцій. Результат перевірки умови з виходу схеми управління полярністю *CO* надходить на трьохстабільну шину *CT*, що управляється сигналом дозволу коду умови $\overline{OECT}$.

Для підключення виходу *CT* СУСЗ до одного з входів мультиплексора МУ БМУ в трирозрядному полі *MS* мікрокоманди, що належить до полів управління БМУ необхідно встановити номер входу мультиплексора (рис. 2.16).

При $\overline{OECT} = 0$ дозволяється вивід коду умови через шину *CT*. Якщо $\overline{OECT} = 1$, то вивід коду умови заборонений, а шина *CT* знаходиться в стані високого опору.

Таблиця 2.17. Управління виходом коду умови *CT* ($\overline{OECT} = 0$)

I_3 I_2 I_1 I_0	$I_5 = I_4 = 0$	$I_5 = 0, I_4 = 1$	$I_5 = 1, I_4 = 0$	$I_5 = I_4 = 1$
0 0 0 0	$(NN \oplus NV) \vee NZ$	$(NN \oplus NV) \vee NZ$	$(MN \oplus MV) \vee MZ$	$(IN \oplus IV) \vee IZ$
0 0 0 1	$\overline{(NN \oplus NV)} \& \overline{NZ}$	$\overline{(NN \oplus NV)} \& \overline{NZ}$	$\overline{(MN \oplus MV)} \& \overline{MZ}$	$\overline{(IN \oplus IV)} \& \overline{IZ}$
0 0 1 0	$NN \oplus NV$	$NN \oplus NV$	$MN \oplus MV$	$IN \oplus IV$
0 0 1 1	$\overline{NN \oplus NV}$	$\overline{NN \oplus NV}$	$\overline{MN \oplus MV}$	$\overline{IN \oplus IV}$
0 1 0 0	NZ	NZ	MZ	IZ
0 1 0 1	$\overline{NZ}$	$\overline{NZ}$	$\overline{MZ}$	$\overline{IZ}$
0 1 1 0	NV	NV	MV	IV
0 1 1 1	$\overline{NV}$	$\overline{NV}$	$\overline{MV}$	$\overline{IV}$
1 0 0 0	$NC \vee NZ$	$NC \vee NZ$	$MC \vee MZ$	$IC \vee IZ$
1 0 0 1	$\overline{NC} \& \overline{NZ}$	$\overline{NC} \& \overline{NZ}$	$\overline{MC} \& \overline{MZ}$	$\overline{IC} \& \overline{IZ}$
1 0 1 0	NC	NC	MC	IC
1 0 1 1	$\overline{NC}$	$\overline{NC}$	$\overline{MC}$	$\overline{IC}$
1 1 0 0	$\overline{NC} \vee NZ$	$\overline{NC} \vee NZ$	$\overline{MC} \vee MZ$	$\overline{IC} \vee IZ$
1 1 0 1	$NC \& \overline{NZ}$	$NC \& \overline{NZ}$	$MC \& \overline{MZ}$	$IC \& \overline{IZ}$
1 1 1 0	$IN \oplus MN$	NN	MN	IN
1 1 1 1	$\overline{IN \oplus MN}$	$\overline{NN}$	$\overline{MN}$	$\overline{IN}$

Таблиця 2.18. Перевірка відношення чисел А і В після виконання операції (А – В)

Відношення	Числа без знаку								
	Стан	$CT = 1$				$CT = 0$			
		I_3	I_2	I_1	I_0	I_3	I_2	I_1	I_0
$A = B$	$Z = 1$	0	1	0	0	0	1	0	1
$A \neq B$	$Z = 0$	0	1	0	1	0	1	0	0
$A \geq B$	$C = 1$	1	0	1	0	1	0	1	1
$A < B$	$C = 0$	1	0	1	1	1	0	1	0
$A > B$	$C \& Z = 1$	1	1	0	1	1	1	0	0
$A \leq B$	$C \vee Z = 1$	1	1	0	0	1	1	0	1
Відношення	Числа в доповню вальному коді								
	Стан	$CT = 1$				$CT = 0$			
		I_3	I_2	I_1	I_0	I_3	I_2	I_1	I_0
$A = B$	$Z = 1$	0	1	0	0	0	1	0	1
$A \neq B$	$Z = 0$	0	1	0	1	0	1	0	0
$A \geq B$	$\overline{N \oplus V} = 1$	0	0	1	1	0	0	1	0
$A < B$	$N \oplus V = 1$	0	0	1	0	0	0	1	1
$A > B$	$[\overline{N \oplus V}] \& \overline{Z} = 1$	0	0	0	1	0	0	0	0
$A \leq B$	$[N \oplus V] \& Z = 1$	0	0	0	0	0	0	0	1

2.3.6. Блок управління переносами

Блок управління переносами (рис. 2.6) формує сигнал вихідного переносу *CO*, який використовується для формування вхідного переносу *CI* АЛБ процесорного елементу. Управління формуванням вихідного переносу здійснюється сигналами $I_{12}, I_{11}, I_5, I_3, I_2, I_1$ поля СУЗС_ІС мікрокоманди як показано в табл. 2.19.

Розряди I_{12}, I_{11} використовуються для присвоєння вхідному переносу значення 0 або 1. Якщо $I_{12}I_{11} = 00$, то $CI = 0$, та якщо $I_{12}I_{11} = 01$, $CI = 1$.

Якщо для формування вихідного переносу окрім розрядів I_{12}, I_{11} (коли $I_{12}I_{11} = 11$) використовують також розряди I_5, I_3, I_2, I_1 поля СУСЗ_ІС МК то вхідному переносу *CI* можна при-

своїти значення розрядів переносу *MC*, *NC* регістрів Рг*M* та Рг*N,* або їх заперечення.

Таблиця 2.19. Управління формуванням сигналу вхідного переносу *CI* АЛБ

I_{12}	I_{11}	I_5	I_3	I_2	I_1	*CO*	Коментар
0	0	*	*	*	*	0	
0	1	*	*	*	*	1	
1	0	*	*	*	*	*CX*	
1	1	0	*	*	*	*NC*	$I_3I_2I_1 \neq 100$
		1	*	*	*	*MC*	$I_3I_2I_1 \neq 100$
		0	1	0	0	$\overline{NC}$	
		1	1	0	0	$\overline{MC}$	

Таким чином, в якості вхідного переносу *CI* використовується одне з наступних джерел – 0, 1, *CX*, *NC*, *MC*, $\overline{NC}$, $\overline{MC}$, що дозволяє легко реалізувати операції додавання та віднімання чисел як звичайної так і подвійної довжини. Вхід *CX* (коли $I_{12}I_{11} = 10$) служить для організації виконання ПЕ К1804ВС2 ряду спеціальних функцій при поєднанні входу *CX* з виходом *Z* ПЕ.

2.3.7. Блок управління зсувами

Блок управління зсувами (рис. 2.6) призначений для організації різних варіантів арифметичних, логічних та циклічних зсувів. Для управляння завданням типу зсувів використовуються сигнали $I_{10} - I_6$ поля СУСЗ_ІС мікрокоманди. Загалом можливі 32 типи зсувів – 16 вліво і 16 вправо (табл. 2.20).

Виводи зсуву *PF*0, *PF*3, *PQ*0, *PQ*3 є трьохстабільними і управляються сигналом $\overline{SE}$. Під час встановлення $\overline{SE} = 0$ зсув дозволений, інакше $\overline{SE} = 1$ зсув заборонений, а всі виводи зсуву знаходяться в стані високого опору.

Таблиця 2.20. Управління зсувами ($\overline{SE} = 0$)

Розряди МК					Модифікація зсуву			Сигнали на виводах ВР2				
1					2			3				
I_{10}	I_9	I_8	I_7	I_6	*MC*	ЗС*B*	ЗС*Q*	*PF*0	*PF*3	*PQ*0	*PQ*3	*MC*
0	0	0	0	0		0	0	*Z*	0	*Z*	0	–
0	0	0	0	1		1	1	*Z*	1	*Z*	1	–
0	0	0	1	0		0	*MN*	*Z*	0	*Z*	*MN*	P*F*0
0	0	0	1	1		1		*Z*	1	*Z*	P*F*0	–
0	0	1	0	0				*Z*	*MC*	*Z*	P*F*0	–
0	0	1	0	1		*MN*		*Z*	*MN*	*Z*	P*F*0	–
0	0	1	1	0		0		*Z*	*0*	*Z*	P*F*0	–

Продовження таблиці 2.16.

1					2	3				
0	0	1	1	1	0	*Z*	*0*	*Z*	P*F*0	P*Q*0
0	1	0	0	0		*Z*	P*F*0	*Z*	P*Q*0	P*F*0
0	1	0	0	1		*Z*	*MC*	*Z*	P*Q*0	P*F*0
0	1	0	1	0		*Z*	P*F*0	*Z*	P*Q*0	–
0	1	0	1	1	*IC*	*Z*	*IC*	*Z*	P*F*0	–
0	1	1	0	0		*Z*	*MC*	*Z*	P*F*0	P*Q*0
0	1	1	0	1		*Z*	P*Q*0	*Z*	P*F*0	P*Q*0
0	1	1	1	0	$IN \oplus IV$	*Z*	$IN \oplus IV$	*Z*	P*F*0	–
0	1	1	1	1		*Z*	P*Q*0	*Z*	P*F*0	–

Продовження таблиці 2.16.

1	1	0	0	1		*MC*	*Z*	P*Q*3	*Z*	P*F*3
1	1	0	1	0		P*F*3	*Z*	P*Q*3	*Z*	–
1	1	0	1	1	0	*MC*	*Z*	0	*Z*	–
1	1	1	0	0		P*Q*3	*Z*	*MC*	*Z*	P*F*3
1	1	1	0	1		P*Q*3	*Z*	P*F*3	*Z*	P*F*3
1	1	1	1	0		P*Q*3	*Z*	*MC*	*Z*	–
1	1	1	1	1		P*Q*3		P*F*3		–

Приклад 2.3. Розробити фрагмент мікропрограми для зсуву вліво слова подвійної довжини. Старша і молодша частини слова записані у регістрах *R*3, *R*4 відповідно.

Виконання завдання:

Операційна схема та мікроалгоритм виконання операції наведена на рис. 2.17, *а*, *б*. Для реалізації зсуву слова подвійної довжини виконаємо два типи зсувів: по-перше – логічний зсув *SL*.16 ($I_{10}..I_6 = 10000$) молодшої частини слова *R*4, під час чого старший розряд, що виходить за межі розрядної сітки, зберігається у розряді *MC* регістру стану *RM*. Далі виконаємо циклічний зсув *SL*.25 ($I_{10}..I_6 = 11001$) старшої частини слова *R*3, під час чого у молодший розряд, звільнений за зсуву, записується вміст розряду *MC* регістру *RM*, в якому в подальшому зберігається старший розряд слова, що вийшов за межі розрядної сітки.

Цифрова діаграма стану регістрів наведена у табл. 2.21.Кодова карта розробленої мікропрограми наведена у табл. 2.22.

Мікропрограма зсуву слів подвійної довжини, налагоджена за допомогою середи модулювання *COMPLEX*, наведена на рис. 2.18.

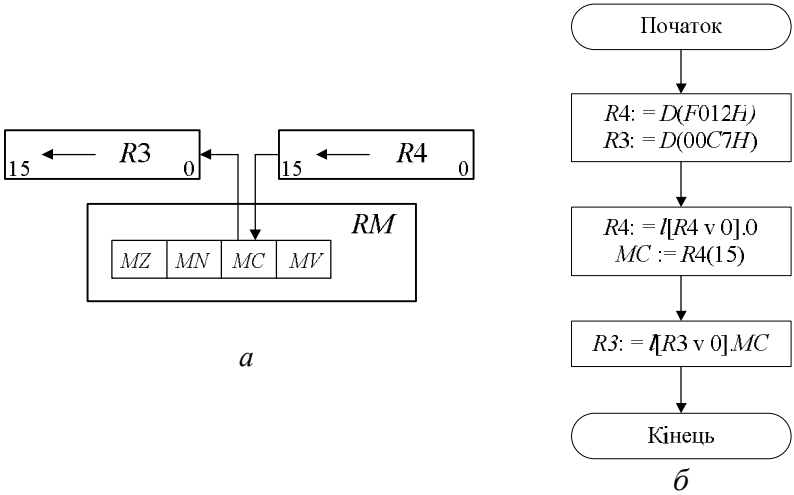


Рис. 2.17. Реалізація зсуву вліво слова подвійної довжини: а – операційна схема; б – мікроалгоритм.

Таблиця 2.21. Цифрова діаграма стану регістрів

№ такту	Вихідний стан			Інформація у приймачі результату, [16]	МС	Мікрооперація
	Шина *D*, [16]	*R*3, [16]	*R*4, [16]			
1	$00C7$	0000	0000	$R3 := 00C7$	*	$R3 := D(007C) \vee 0$
2	$F012$	$00C7$	0000	$R4 := F012$	*	$R4 := D(F012) \vee 0$
3	****	$00C7$	$F012$	$R4 := E024$	1	$R4 := SLL(R4 \vee 0)$
4	****	0070	$E024$	$R3 := 018F$	1	$R3 := S.25(R3 \vee 0)$
		$018F$	$E024$			Результат

Примітка: *SLL, S.*25 – мнемонічне позначення зсувів логічного вправо та циклічного відповідно.

Таблиця 2.22 Кодова карта мікропрограми зсуву слів подвійної довжини

№ такту	Адреса [16]	Константа БОД		БОД													
				ВС1 (АЛБ)				ВР2 (СУСЗ)									
		D [16]	$\overline{OED}$	АЛБ_МІ [2]	A [16]	B [16]	$\overline{OEY}$	СУСЗ_МІ [2]	$\overline{EC}$	$\overline{EZ}$	$\overline{EN}$	$\overline{EV}$	$\overline{CEN}$	$\overline{CEM}$	$\overline{OECT}$	$\overline{SE}$
1	0000	$00C7$	0	011 011 101	3	3	1	00 00000 000000	0	0	0	0	1	1	1	1
2	0001	$F012$	0	011 011 101	4	4	1	00 00000 000000	0	0	0	0	1	1	1	1
3	0002	****	1	111 011 100	4	4	1	00 10000 000000	0	0	0	0	1	1	1	0
4	0003	****	1	111 011 100	3	3	1	00 11001 000000	0	0	0	0	1	1	1	0

```
COMPLEX
Файл   Редактор   Выполнить   Протокол   Помощь   Опции   Выход

Исходный файл :D_LEFT.PMK  Страница: 1.1
```

Адр	Конст. БОД БМУ	OED	Б О Д В С 1 MI*	A	B	OEY	В Р 2 MI*	C	Z	N	V	OECT	CEN	CEM	SE
000	00C7	0	011.011.101	3	3	1	00.00000.000000	0	0	0	0	1	1	1	1
001	F012	0	011.011.101	4	4	1	00.00000.000000	0	0	0	0	1	1	1	1
002	0000	1	111.011.100	4	4	1	00.10000.000000	0	0	0	0	1	1	1	1
003	0000	1	111.011.100	3	3	1	00.11001.000000	0	0	0	0	1	1	1	1
004	0000	1	001.000.000	0	0	1	00.00000.000000	0	0	0	0	1	1	1	1

```
Исходный файл :D_LEFT.PMK  Страница: 1.2
```

Адр	БМУ и БПП PAA,PAB MSA	MSB	EWA	EWB	ВУ4 MI*	CCE	COM	CI	RLD	MS	ВН1 MI*	EINS	ПАВ EV	ОП,ВУ,РА ВУ I	O	LCK	IA	ОП R	W	PrA EWH	EWL
000	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
001	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
002	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
003	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
004	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1

```
Нажмите любую клавишу для окончания
```

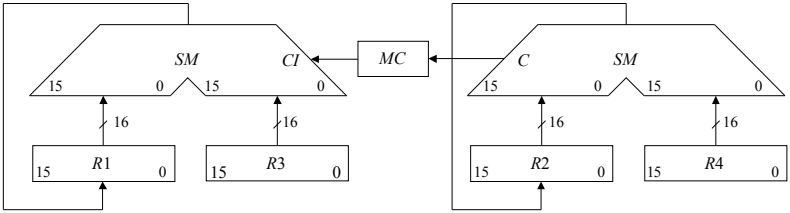
```
Исходный файл :D_LEFT_A.ASM  Страница: 1

accept r3:00c7h
accept r4:0f012h
{or sll,r4,r4,z;}
{or sl.25,r3,r3,z;}
{}
```

Рис. 2.18.Мікропрограма зсуву вліво слова подвійної довжини

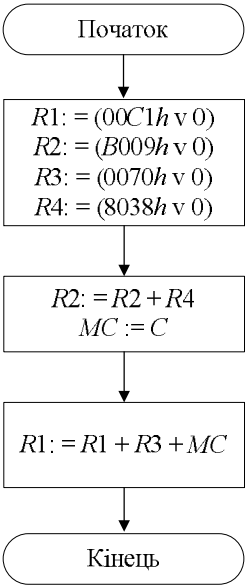
Приклад 2.4. Розробити фрагмент мікропрограми для додавання слів подвійної довжини. Доданки записані у регістрах *R*1, *R*2 та *R*3, *R*4 відповідно.

Виконання завдання:

Операційна схема та мікроалгоритм виконання операції наведена на рис. 2.19, *а*, *б*.



а



б

Рис. 2.19. Операція додавання слів подвійної довжини: *а* – операційна схема; *б* – мікроалгоритм

Для реалізації додавання слів подвійної довжини виконаємо послідовне додавання молодших і старших частин слова із поширенням переносу із молодшої частини результату у старшу. Для реалізації поширення переносу під час додавання молодших частин слів збережемо ознаку переносу у розряді *MC* регістру стану *RM*:

$$R2 := R2 + R4\ ;\ LOAD\ MC,$$

під час додавання старших частин, подамо значення переносу на вхід *CI* АЛБ:

$$R1 := R1 + R3 + CI,\ \text{де}\ CI := MC.$$

Цифрова діаграма стану регістрів наведена у табл. 2.23.Кодова карта розробленої мікропрограми наведена у табл. 2.24.

Мікропрограма додавання слів подвійної довжини налагоджена за допомогою середи модулювання *COMPLEX* наведена на рис. 2.20.

2.3.8. Виконання операції нормалізації

ІС К1804ВР2 забезпечує виконання операції нормалізації чисел звичайної і подвійної довжини, як для ПЕ К1804ВС1, так і для ПЕ К1804ВС2.

Поєднання виводів мікросхем К1804ВР2, К1804ВС1 і К1804ВУ4 показано на рис. 2.7.

В мікросхемі К1804ВС1 однією з ознак закінчення нормалізації під час виконання чергової мікрооперації служить встановлення сигналу на виході *CO* – старшої секції результату (під час реалізації обчислень у МПС із словами довжиною більш ніж чотири розряди ПЕ поєднуються у ланцюги, див розділ 2.4 рис. 2.22). При цьому на виході *CO* формується сигнал, як результат виконання операції ВИКЛЮЧНЕ АБО над двома старшими розрядами.

Другою ознакою, що інформує про закінчення нормалізації служить сигнал на виході *OVR*, що є результатом операції ВИКЛЮЧНЕ АБО над двома середніми розрядами старшої секції результату. Ця ознака встановлюється за один такт до закінчення операції. Сигнали *CO* або *OVR* через блок перевірки умови СУСЗ виставляються на виході *CT* К1804ВС2 та надходять на вхід умови *CC* БМУ ІС К1804ВУ4 (див. розділ 2.5, рис. 2.27).

Таблиця 2.23. Цифрова діаграма стану регістрів

№ такту	Вихідний стан					Інформація у приймачі результату, [16]	МС	Мікрооперація
	Шина ***D***, [16]	***R*1**, [16]	***R*2**, [16]	***R*3**, [16]	***R*4**, [16]			
1	$00C1$	0000	0000	0000	0000	$R1 := 00C1$	*	$R1 := D(00C1) \vee 0$
2	0009	$00C1$	0000	0000	0000	$R2 := 0009$	*	$R2 := D(0009) \vee 0$
3	0070	$00C1$	0009	0000	0000	$R3 := 0070$	*	$R3 := D(0070) \vee 0$
4	8038	$00C1$	0009	0070	0000	$R4 := 8038$	*	$R4 := D(8038) \vee 0$
4	****	$00C1$	0009	0070	8038	$R2 := 3041$	1	$R2 := R2 + R4 + CI$
5	****	$00C1$	3041	0070	8038	$R1 := 00CB$	*	$R1 := R3 + R1 + 1(MC)$
		$00CB$	3041					Результат

Таблиця 2.24 Кодова карта мікропрограми додавання слів подвійної довжини

№ такту	**Адреса** [16]	**Константа БОД**		**БОД**													
				ВС1 (АЛБ)				**ВР2 (СУСЗ)**									
		D [16]	$\overline{OED}$	АЛБ_МІ [2]	A [16]	B [16]	$\overline{OEY}$	СУСЗ_МІ [2]	$\overline{EC}$	$\overline{EZ}$	$\overline{EN}$	$\overline{EV}$	$\overline{CEN}$	$\overline{CEM}$	$\overline{OECT}$	$\overline{SE}$
1	0000	00*C*1	0	011 011 101	1	1	1	00 00000 000000	0	0	0	0	1	1	1	1
2	0001	0009	0	011 011 101	2	2	1	00 00000 000000	0	0	0	0	1	1	1	1
3	0002	0070	0	011 011 101	3	3	1	00 00000 000000	0	0	0	0	1	1	1	1
4	0003	8038	0	011 011 101	4	4	1	00 00000 000000	0	0	0	0	1	1	1	1
5	0004	****	1	011 000 001	4	2	1	00 00000 000110	0	0	0	0	1	0	1	1
6	0005	****	1	011 000 001	3	1	1	11 00000 100000	0	0	0	0	1	1	1	1

COMPLEX

Файл Редактор Выполнить Протокол Помощь Опции Выход

Исходный файл :D_SUM.PMK Страница: 1.1

Адр	Конст. БОД БМУ	Конст. БОД OED	Б О Д В С 1 MI*	A	B	OEY	В Р 2 MI*	C	Z	N	V	OECT	CEN	CEM	SE
000	00C1	0	011.011.101	1	1	1	00.00000.000000	0	0	0	0	1	1	1	1
001	B009	0	011.011.101	2	2	1	00.00000.000000	0	0	0	0	1	1	1	1
002	0070	0	011.011.101	3	3	1	00.00000.000000	0	0	0	0	1	1	1	1
003	8038	0	011.011.101	4	4	1	00.00000.000000	0	0	0	0	1	1	1	1
004	0000	1	011.000.001	4	2	1	00.00000.000110	0	0	0	0	1	1	0	1
005	0000	1	011.000.001	3	1	1	11.00000.100000	0	0	0	0	1	1	1	1
006	0000	1	001.000.000	0	0	1	00.00000.000000	0	0	0	0	1	1	1	1

Исходный файл :D_SUM.PMK Страница: 1.2

Адр	РАА,РАВ MSA	MSB	EWA	EWB	БМУ и БПП ВУ4 MI*	CCE	COM	CI	RLD	MS	ВН1 MI*	EINS	ПАВ EV	ОП,ВУ,РА ВУ I	O	LCK	IA	ОП R	W	РгА EWH	EWL
000	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
001	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
002	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
003	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
004	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
005	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
006	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1

Нажмите любую клавишу для окончания

```
Исходный файл :D_SUM_A.ASM  Страница: 1

accept r1:00c1h
accept r2:0b009h
accept r3:0070h
accept r4:08038h
{add r2,r2,r4,z;load rm,flags;}
{add r1,r1,r3,rm_c;}
{}
```

Рис. 2.20. Мікропрограма додавання слів подвійної довжини

Оскільки розглянуті ознаки записуються в регістр стану з запізненням на один такт, то при використанні в якості ознаки закінчення нормалізації сигналу сформованого на виході *C*4 необхідно останнім кроком нормалізації виконати зсув вбік молодших розрядів. Такий зсув виконується СУСЗ спеціальною операцією зсуву за надходження на входи $I_{10} - I_6$ відповідних кодів: $I_{10}, I_9, I_8, I_7, I_6 = 01001$ – нормалізація чисел подвійної довжини або $I_{10}, I_9, I_8, I_7, I_6 = 00010$ – нормалізація чисел звичайної довжини.

В процесорному елементі К1804ВС1 не передбачена спеціальна мікрооперація нормалізації. Нормалізація виконується з застосуванням мікрооперації зсуву. Для виконання функції елемента ВИКЛЮЧНЕ АБО, який відсутній в мікросхемі К1804ВС1, в СУСЗ передбачена спеціальна операція ($MN \oplus IN$), що виконується із знаковими розрядами числа сформованими у попередньому такті обчислень (*MN*) і у поточному такті (*IN*).

2.3.9. Реалізація переривань

Мікросхема К1804ВР2 може використовуватися для організації переривань як на програмному, так і на мікропрограмному рівні. При виконанні переривань відбувається передавання ознак стану з регістру Рг*M* (під час переривань програмного рівня) або з регістрів Рг*N*, Рг*M* (під час переривань мікропрограмного рівня) в зовнішні регістри. Таке передавання ознак стану реалізується через двоспрямовану шину *Y* (див. табл. 2.16).

Після закінчення переривання в Рг*M* і Рг*N* повинен бути відновлений стан, що передував перериванню. Вміст Рг*N,* що передував перериванню, спочатку записується в Рг*M*, а далі за запису в Рг*M* ознак стану, що передували перериванню, вміст Рг*M* передається в Рг*N*.

Організація системи переривань в мікропроцесорних системах розглянута у розділі 2.7

2.4. Блоки обробки даних

Блоки обробки даних будуються на основі процесорних елементів, наприклад, ІС К1804ВС1. Нарощування довжини розрядної сітки процесора забезпечується поєднанням декількох ПЕ, як показано на рис. 2.21. для випадку, коли $n = 16$. Шини *AA*, *AB*, *I* і *CLK*, які умовно не зображені, підключаються паралельно до кожної ІС ВС1, аналогічно шині *OE*. З ціллю зменшення часу розповсюдження переносу використовують ІС К1804ВР1, умовне графічне позначення якої показано на рис. 2.22. В цьому випадку (коли $n = 16$) ланцюги розповсюдження переносу реалізують, відповідно схемі зображеній на рис. 2.23.

Мікросхеми ВС1 мають відповідні виходи *P* і G, що дозволяє при побудові процесорів з довжиною слова $n > 16$ здійснювати каскадне підключення цих мікросхем (рис. 2.23).

Часові параметри ІС К1804ВР1 приведені в табл. 2.25.

Таблиця 2.25. Парамети ІС К1804ВР1.

Час затримки сигналів	Значення, нс
від *CI* до *CO*	10
від *P*0 – *P*3 до *P*	8,5
від *P*0 – *P*3, *G*0 – *G*3 до *CX*, *CY*, *CZ*	7
від *P*0 – *P*3, *G*0 – *G*3 до *G*	9

Виходи ознак і зсувачів підключають до однойменних входів ІС К1804ВР2 (рис. 2.9). Відповідні виходи цієї ІС забезпечують подачу на блок управління різних ознак і вхідного переносу для молодшої секції процесору.

Для управління роботою шістнадцятирозрядного операційного блоку у структурі мікрокоманди задіяні наступні поля (рис. 2.8):

що управляють ІС К1804ВС1 – АЛБ_МІ, *AB*, *AA*, $\overline{OE}$;

що управляють ІС К1804ВР2 – СУСЗ_МІ, $\overline{OECT}$, $\overline{EZ}$, $\overline{EN}$, $\overline{EC}$, $\overline{EV}$, $\overline{CEN}$, $\overline{CEM}$, $\overline{OEY}$, $\overline{SE}$.

Дані приймаються в операційний блок по шині *D*, а видаються по шині *Y* ІС К1804ВС1. Двоспрямована шина *Y* ІС К1804ВР2 дозволяє, наприклад, зберегти в стеку і завантажити ознаки при роботі з підпрограмами.

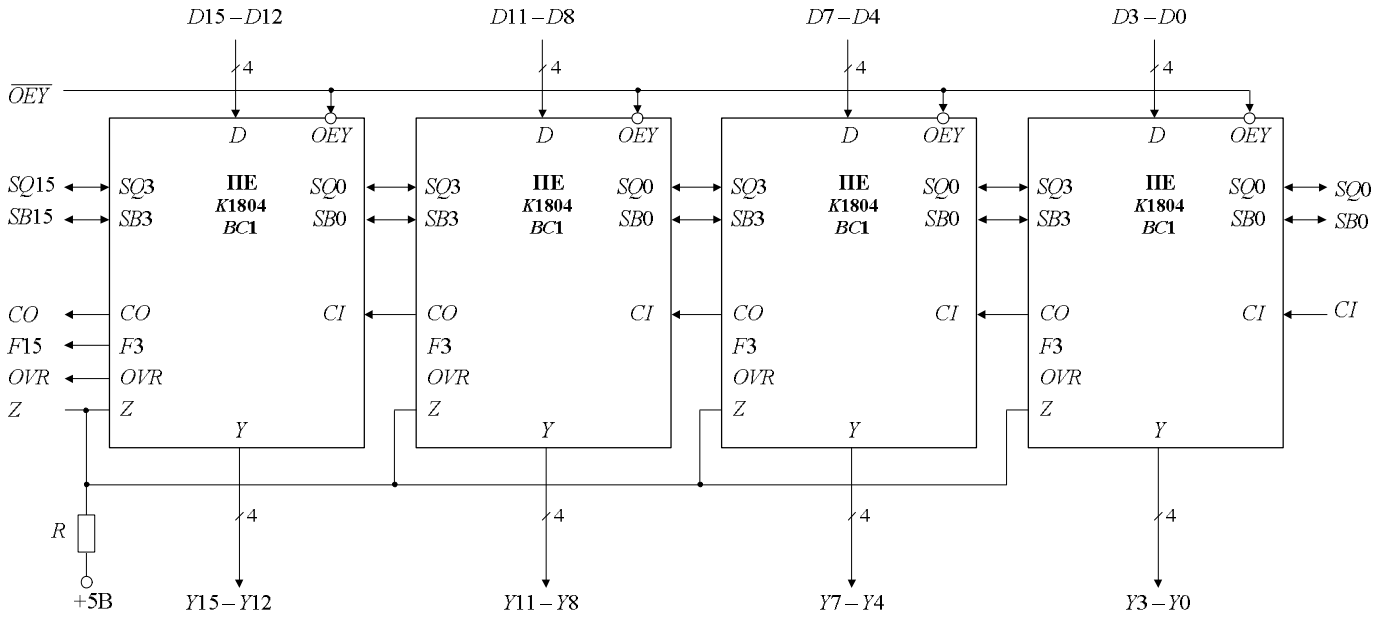


Рис 2.21. Структурна схема шістнадцятирозрядного ПЕ

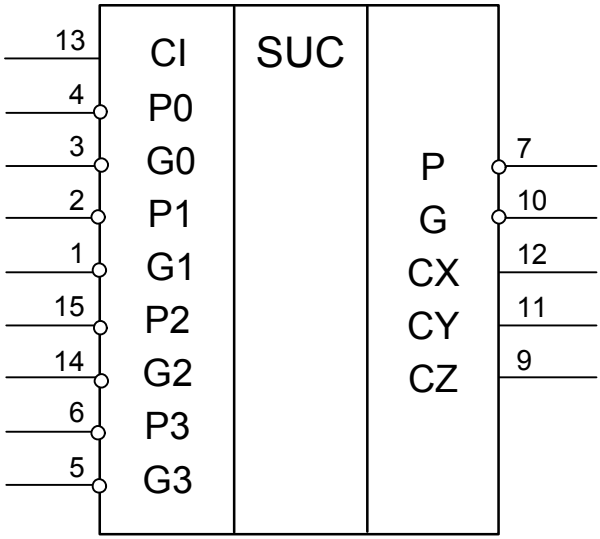


Рис. 2.22 Умовне графічне позначення ІС К1804ВР1

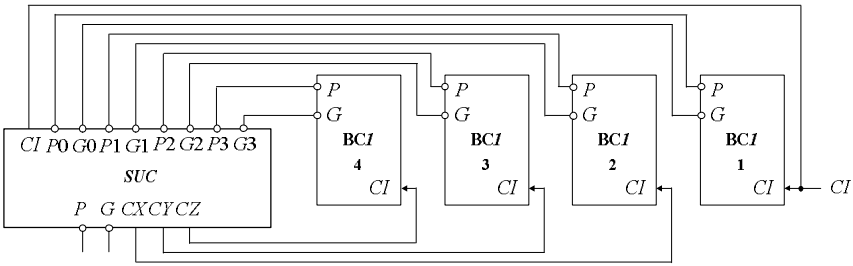


Рис. 2.23. Організація ланцюгів розповсюдження переносу шістнадцяти-розрядного ПЕ

Приклад 2.5. Розробити мікропрограму обчислення заданої функції для шістнадцятирозрядного блоку обробки даних.

$$F = 2X3 \,\&\, (X2 - X1)/2$$

Вихідні дані: $X1 = -3$, $X2 = 15$, $X3 = 7$.

Вважатимемо, що для подання операндів використовується доповнювальний код. Мікроалгоритм обчислення функції F показаний

на рис. 2.24. В коментарях до кожної операторної вершини наведені значення управляючих сигналів ІС, *AA*, *BB*, *CI*, *SB*15, *SB*0, *SQ*15, *SQ*O, що забезпечують виконання відповідної мікрооперації.

Рис. 2.24. Мікроалгоритм обчислення функції

Послідовність мікрокоманд, що релізують заданий мікроалоритм, відображена у табл. 2.25.

Розглянемо мікрооперацію $[RQ - D(X1)]/2 \to R0$, що виконується в другому такті. Джерелами даних, що надходять на входи *S* і *R* АЛБ (рис. 2.1), є регістр *RQ* і дані на вхідній шині *D*. Відповідно до табл. 2.4. визначаємо розряди $I_2 I_1 I_0$ поля АЛБ_МІ мікрокоманди $I_2 I_1 I_0 = 110$. Мікрооперація віднімання, що здійснюється у цьому такті, визначається розрядами $I_5 I_4 I_3 = 001$ поля АЛБ_МІ мікрокоманди (табл. 2.3). при цьому встановлюється *C1*=1, що відповідає розрядам $I_{12} I_{11} = 01$ поля СУСЗ_МІ мікрокоманди (табл. 2.19). Результат, отриманий на виході *Y* АЛБ записується в регістр НОЗП за адресою, виставленою на шині *AB*. За умовою завдання результат має бути записаний у регістр *R*0, тому у полі *B* АЛБ_МІ мікрокоманди (табл. 2.3) розміщується адреса відповідного регістру у шістнадцятирічному поданні B = 0. При цьому кодуємо розряди $I_8 I_7 I_6$ поля АЛБ_МІ мікрокоманди (табл. 2.2), враховуючи те, що результат

мікроопервції має бути зсунутий на один розряд вправо (що відповідає діленню на два за умовою завдання), отримуємо $I_8 I_7 I_6 = 101$.

При арифметичному зсуві вправо розряд, що звільнився, заповнюється знаком числа. Оскільки зсув відбувається на зсувачі ЗСВ, то з урахуванням можливого переповнювання розрядної сітки необхідно забезпечити $SB15 = F15 \oplus OVR$ (де, $F15$ – знак результату, OVR – ознака переповнювання).

Відповідну комутацію виводів зсувача та формування сигналу переносу CO забезпечує ІС ВР2. Подача нуля на вхід переносу ($CI = 0$) забезпечується встановленням розрядів поля СУСЗ_МС мікрокоманди $I_{12} I_{11} = 00$, подача одиниці ($CI = 1$) – встановленням $I_{12} I_{11} = 01$ поля СУСЗ_МІ мікрокоманди.

Якщо зсув не виконується, виводи ВР2, пов'язані із зсувачем, відключаються сигналом $\overline{SE} = 1$, що відповідає розрядам $I_{12} I_{11} = 01$ поля СУСЗ_МІ мікрокоманди (табл. 2.19). Під час виконання зсувів встановлюєься $\overline{SE} = 0$.

Різні типи зсувів кодуються розрядами $(I_{10}..I_6)$ поля СУСЗ_МІ мікрокоманди (табл. 2.19) відповідно до табл. 2.16. Арифметичним зсувам відповідають наступні управляючі сигнали, що надходять на входи ВР2: зсув вправо $I_{10}..I_6 = 01110$, вліво – $I_{10}..I_6 = 10010$.

Таблиця 2.25. Кодова карта

№ такту	Адреса [16]	Константа БОД		БОД												
				ВС1 (АЛБ)				ВР2 (СУСЗ)								
		D [16]	$\overline{OED}$	АЛБ_МІ [2]	*A* [16]	*B* [16]	$\overline{OEY}$	СУСЗ_МІ [2]	$\overline{EC}$	$\overline{EZ}$	$\overline{EN}$	$\overline{EV}$	$\overline{CEN}$	$\overline{CEM}$	$\overline{OECT}$	$\overline{SE}$
1	0000	000*F*	0	000 011 111	*	*	1	00 00000 000000	0	0	0	0	1	1	1	1
2	0001	*FFFD*	0	101 001 110	*	0	1	01 01110 000000	0	0	0	0	1	1	1	0
3	0002	0007	0	111 011 111	*	1	1	00 10010 000000	0	0	0	0	1	1	1	0
4	0003	****	1	011 100 001	0	1	1	00 00000 000000	1	1	1	1	1	1	1	1

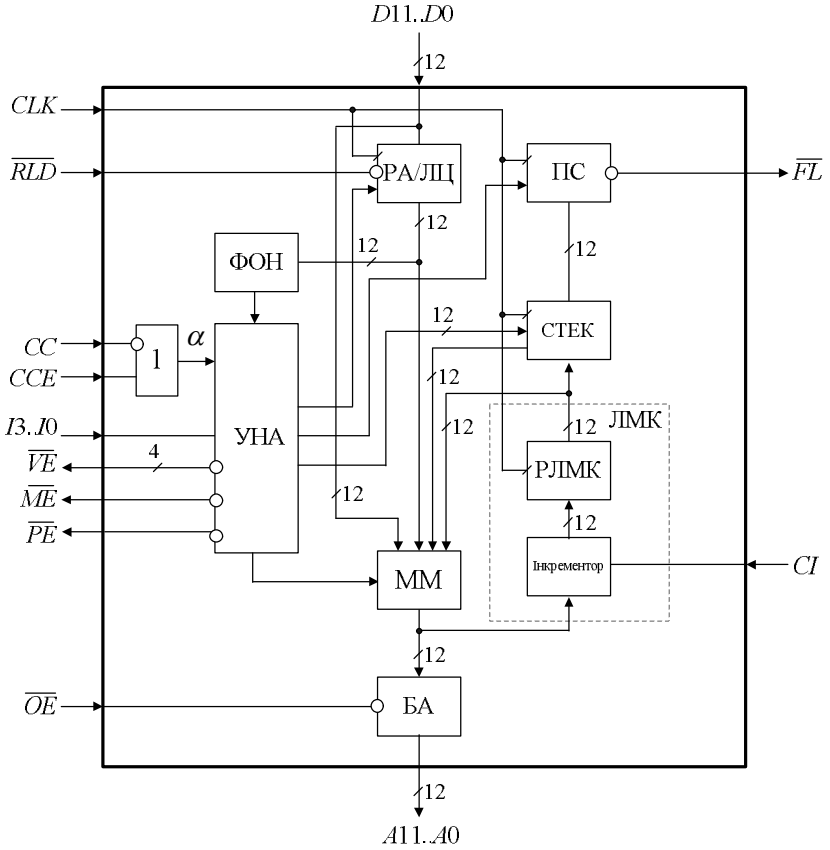
Таблиця 2.26. Цифрова діаграма стану регістрів

№ такту	Вихідний стан				Інформація у приймачі результату, [16]	Мікрооперація
	Шина ***D***, [16]	***R*0**, [16]	***R*1**, [16]	***RQ***, [16]		
1	000*F*	****	****	****	$RQ := 000F$	$RQ := D(000F) \vee 0$
2	*FFFD*	****	****	000*F*	$R0 := 0009$	$R0 := SRA[RQ - D(FFFD) - 1 + 1(CI)]$
3	0007	0009	****	000*F*	$R1 := 000E$	$R1 := SLA[D(0007) \vee 0]$
4	****	0009	000*E*	000*F*	$R1 := 0008$	$R1 := R1 \vee R0$
			0008			Результат

Примітка: *SRA*, *SLA* – мнемонічне позначення зсувів арифметичних вправо та вліво відповідно.

2.5. Блоки мікропрограмного управління

Мікросхема К1804ВУ4 призначена для управління послідовністю вибірки мікрокоманд з пам'яті мікрокоманд (ПМК).Структура ІС К1804ВУ4 зображена на рис. 2.25.



Ри

с 2.25. Структурна схема ІС К1804ВУ4

Мікросхема К1804ВУ4 складається з наступних функціональних частин:

УНА – схема управління вибіркою адреси наступної мікрокоманди;

РА/ЛЦ – регістр адреси/лічильник циклів;

ЛМК – лічильник мікрокоманд;
ПС – покажчик стека;
РЛМК – регістр лічильника мікрокоманд;
ММ – мультиплексор мікрокоманди;
ФОН – формувач ознаки нуля;
БА – буфер адреси.

Умовне графічне позначення ІС К1804ВУ4 показано на рис. 2.26, а призначення виводів наведене в табл. 2.26.

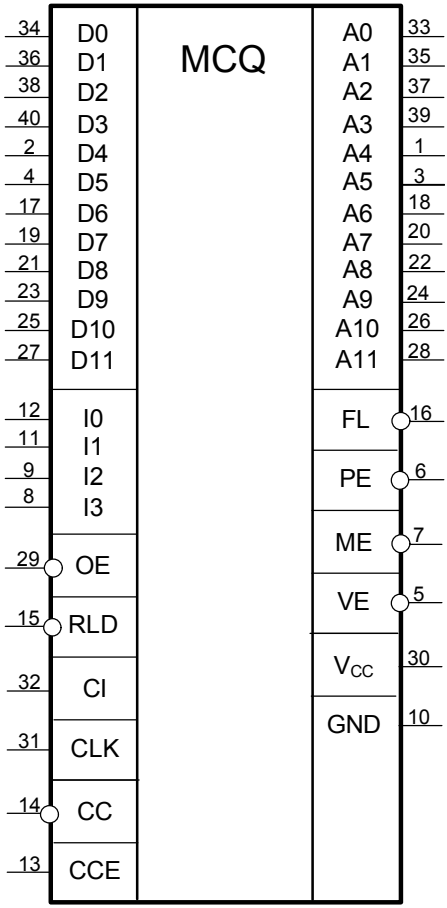


Рис. 2.26. Умовне графічне позначення ІС К1804ВУ4

ІС К1804ВУ4 виконує функції формувача адреси наступної мікрококманди (ФАМ) і забезпечує формування дванадцятирозрядних адрес мікрокоманд у ПМК. Внаслідок чого ємність ПМК складає 4К 83-розрядних слів МК.

Виходи *A* адреси мікрокоманди (рис. 2.26) мають три стани (0, 1 і високоомний стан), що дозволяє, за необхідності, забезпечити зовнішній доступ до входів адреси ПМК, наприклад, для тестової перевірки послідовності мікрокоманд, виходу на початкову мікропрограму ініціалізації системи і таке інше.

Таблиця 2.26. Функціональне призначення виводів ІС ВУ4

Номер виводу	Позначення виводу	Функціональне призначення
33, 35, 37, 39, 1, 3, 18, 20, 22, 24, 26, 28	$A11 - A0$	Виходи адреси мікрокоманд
34, 36, 38, 40, 2, 4, 17, 19, 21, 23, 25, 27	$D11 - D0$	Входи адреси розгалуження
5, 6, 7	*VE*, *PE*, *ME*	Виходи дозволу підключення до входів *D* Буферу *V*, Буферу *P* та Буферу *M* відповідно
12, 11, 9, 8	$I3 - I0$	Входи коду інформаційного слова мікроінструкції переходу
10	*GND*	Вивід живлення (загальний)
13	*CC*	Вхід умови
14	*CCE*	Вхід дозволу умови
15	*RLD*	Вхід дозволу запису в РА/ЛС
16	*FL*	Вихід ознаки заповнення стеку
29	*OE*	Вхід дозволу видачі адреси
30	*VCC*	Вихід живлення $(+5B)$
31	*CLK*	Вхід синхросигнала
32	*CI*	Вхід переносу в лічильник мікрокоманд

Принцип формування адреси наступної мікрокоманди

Спосіб формування адреси мікрокоманди визначається зовнішніми управляючими сигналами на входах $I3 - I0$, $\overline{CC}$, CCE (рис. 2.25) і внутрішнім сигналом, який формується у схемі ФОН.

Схема УНА, що є перетворювачем кодів, виробляє набір внутрішніх управляючих сигналів і зовнішні сигнали $\overline{PE}$, $\overline{VE}$ і $\overline{ME}$. Внутрішні управляючи сигнали виконують настройку блоків ІС схеми на виконання необхідний функцій. Зовнішні управляючі сигнали використовують для підключення до входів $D11 - D0$ ІС відповідних джерел інформації.

Мультиплексор ММ управляється внутрішніми сигналами, що надходять зі схеми УНА, та виконує функцію вибору одного з чотирьох джерел надходження адреси наступної мікрокоманди.

Таким чином, джерелами адреси наступної мікрокоманди є:

– входи $D11 - D0$,
– регістр адреси/лічильник циклів – РА/ЛЦ,
– СТЕК,
– лічильник мікрокоманд – ЛМК.

Обрана адреса через БА видається на виходи A за встановлення управляючого сигналу $\overline{OE} = 0$.

Дані в дванадцятирозрядний регістр РА/ЛЦ надходять з входів $D11 - D0$ та записуються за додатним перепадом синхросигналу CLK, якщо виконується певна управляюча мікроінструкція, що надходить на входи $I3 - I0$ ІС. Під час запису даних у РА/ЛЦ встановлюється управляючий сигнал $\overline{RLD} = 0$, при цьому дані у РА/ЛЦ можуть бути записані і незалежно від виконуваної мікроінструкції.

При виконанні відповідних мікроінструкцій ($I3 - I0$) вміст РА/ЛЦ зменшується на одиницю за додатним сигналом CLK, що дозволяє використовувати РА/ЛЦ в якості лічильника циклів. Ознака того, що вміст лічильника циклів дорівнює нулю – РА/ЛЦ = 0, формується у схемі ФОН.

Таким чином, у РА/ЛЦ надходить інформація з зовнішніх входів D (при умові встановлення $\overline{RLD} = 0$), і у подальшому використовується або в якості адреси наступної мікрокоманди, або як кількість повторів циклів.

Лічильник мікрокоманд складається з дванадцятирозрядного РЛМК та ІНКРЕМЕНТОРА, що є комбінаційною схемою додавання вхідного переносу CI до вхідного слова. Будь-яка адреса, з виходу мультиплексора ММ записується за додатним перепадом CLK в регістр РЛМК, під час встановлення сигналу $CI = 0$, або збільшеною на одиницю, під час встановлення сигналу $CI = 1$. Таким чином на-

прикінці кожного такту у ЛМК формується адреса наступної мікрокоманди у вигляді $A_{i+1} = A_i + 1$, яка у наступному такті при виконанні лінійних переходів є джерелом адреси наступної мікрокоманди.

Стек складається з покажчика стеку (ПС) і накопичувача (СТЕК), що має п'ять регістрів (глибина стеку дорівнює п'яти). Схема ПС є реверсивним лічильником, який управляється внутрішніми сигналами і змінює свій стан за додатним перепадом *CLK*.

Стек організований за принципом *LI/FO* (*Last Input / First Output* – останній прийшов / перший обслугований). Схема ПС вказує на регістр, у який за останнім зверненням здійснювався запис даних.

Залежно від виконуваних мікроінструкцій (*I*3 –*I*0) стек працює в наступних режимах:

– *очищення стека* – встановлення покажчика стеку в нуль (ПС = 0);

– *збереження інформації* – ПС не змінює свого стану, з відповідного регістру накопичувача СТЕК (визначеному ПС) здійснюється зчитування даних;

– *завантаження стеку* – за додатним перепадом *CLK* значення ПС збільшується на одиницю в послідовності 0, 1, 2, 3, 4, 5, після чого здійснюється запис інформації в регістр накопичувача СТЕК; при заповненні стека (ПС = 5) встановлюється сигнал $\overline{FL} = 0$ і при подальшому завантаженню стеку ПС не змінює свого стану;

– *виштовхування із стеку* – відбувається зчитування інформації із стеку і зменшення ПС на одиницю в послідовності 5, 4, 3, 2, 1, 0; при встановленні ПС = 0 операція виштовхування із стеку приводить до зчитування невизначеної інформації, при цьому ПС не змінює свого стану.

ІС К1804ВУ4 дозволяє реалізувати 16 мікроінструкцій, які застосовуються для розгалуження мікропрограм і забезпечують:

– лінійні переходи – формування наступної адреси (інкремент);
– багаторазове повторення однієї й той самої адреси;
– умовні та безумовні переходи;
– організацію циклів;
– умовні та безумовні виклики мікропідпрограм.

Мікроінструкції є безумовними і умовними, тобто виконуваними при виконанні певної умови. В залежності від сигналів на вхо-

дах $\overline{CC}$ і CCE формується ознака виконання умови $\alpha = \overline{CC} \vee CCE$ (рис. 2.25). Якщо умова виконується встановлюється $\alpha = 1$, інакше $\alpha = 0$. Таким чином умова виконується, якщо $\overline{CC} = 0$ або $CCE = 1$. Незалежно від значення сигналу на вході $\overline{CC}$, можна примусово забезпечити виконання умови $\alpha = 1$, встановив $CCE = 1$, тобто забезпечити безумовний перехід.

Характер переходів під час виконання мікроінструкцій наведено у табл. 2.27.

На базі ІС К1804ВУ4, призначеної для формування та управління послідовністю вибору мікрокоманд із ПМК можна побудувати БМУ, структурна схема якого зображена на рис. 2.27.

БМУ складається з наступних функціональних частин:

ФАМ – схема формування вибіркою адреси наступної мікрокоманди, реалізована на ІС ВУ4;
ПМК – пам’ять мікрокоманд;
РМК – регістр мікрокоманди;
МУ – мультиплексор вибору логічних умов;
ІНВЕРТОР – інвертор;
три буфери джерел адрес мікрокоманд:
Буфер *M*, – буфер регістра адреса,
Буфер *V*, – буфер регістра вектора переривань,
Буфер *P* – буфер регістра константи;
ШАР – дванадцяти розрядна шина адреси розгалуження.

Для управління БМУ у структурі мікрокоманди відведено двацять три розряди (рис. 2.28):

P – Дванадцятирозрядна частина поля констант МК (поле *P*), що містить адресу переходу або константу, наприклад, кількість повторень циклів;
ФАМ_ІС – чотирирозрядне інформаційне слово, що визначає мікроінструкцію переходу;
CCE – дозвіл аналізу умови;
COM – інвертування сигналу логічної умови;
CI – дозвіл формування адреси наступної МК;
$\overline{RLD}$ – дозвіл запису константи з поля Р МК в РА/ЛЦ ФАМ;
MS – трирозрядне поле управління вибором входу МУ.

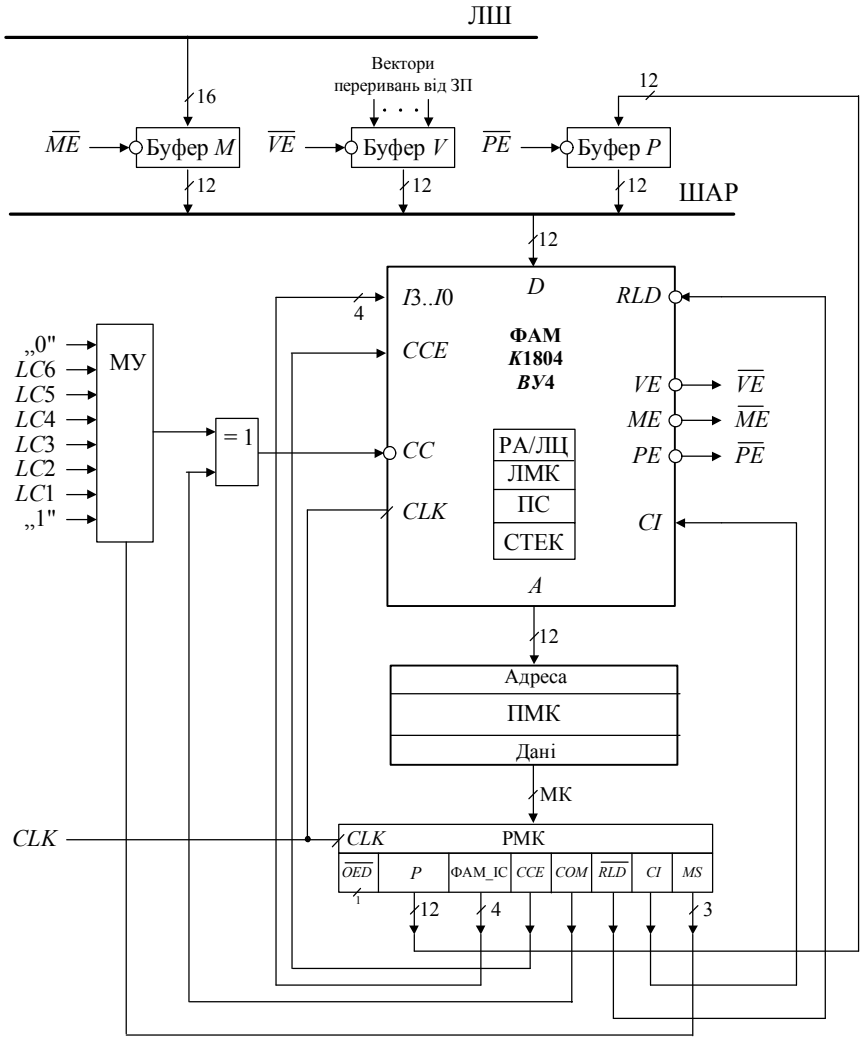


Рис 2.27. Структурна схема БМУ

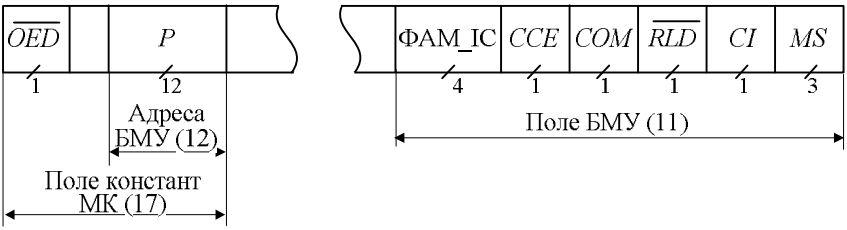


Рис. 2.28. Поля мікрокоманди, які використовуються для управління блоком мікропрограмного управлінн

Принцип функціонування БМУ

Буфери *M*, *V* і *P*, що мають виходи з трьома станами, призначені для видачі на ШАР дванадцятирозрядних адрес. Сигнали $\overline{ME}$, $\overline{VE}$ і $\overline{PE}$ що формує УНА (рис. 2.26) відкривають Буфери *M*, *V* і *P*, відповідно – $\overline{ME}=0$, $\overline{VE}=0$ і $\overline{PE}=0$. У поточному машинному такті може бути відкритий тільки один з буферів. Таким чином, якщо джерелом адреси наступної мікрокоманди у ІС ВУ4 є входи $D11$ – $D0$, то інформація на ці входи видається з одного з буферів *M*, *V* або *P* по ШАР.

Зазвичай Буфер *M* та сигнал $\overline{ME}=0$ використовується для прийому початкової адреси *M* мікропрограми, який визначається полем коду операції команди та надходить у БМУ з локальної шини МПС.

Через Буфер *V* ($\overline{VE}=0$) здійснюється прийом початкової адреси *V* мікропрограми обслуговування переривання від зовнішнього пристрою за певним вектором.

Сигнал $\overline{PE}$ управляє Буфером *P*, через який з поля *P* регістра мікрокоманди на шину ШАР, а далі на шину *D* схеми ФАМ поступає адреса розгалуження або параметр циклу.

Зовнішні логічні умови $LC1-LC6$ надходять на вхід МУ, що комутує ці умови, а також значення 0 та 1 на вихід. З виходу МУ логічний сигнал надходить на вхід елементу ВИКЛЮЧНЕ АБО, який виконує функцію керованого інвертора, і далі на вхід $\overline{CC}$ схеми ФАМ. Інвертуванням сигналу умови управляє сигнал *COM* поля БМУ мікрокоманди. При встановленні сигналу $COM=1$ логічна умова інвертується.

Слід зазначити, що зовнішні логічні умови, що надходять на входи $LC1 - LC6$ мультиплексора МУ, формуються на виході *CT* БОД (рис. 2.6).

Таким чином, БМУ дозволяє реалізувати такі типові управляючі конструкції, як безумовні і умовні переходи, цикли, мікропідпрограми.

Безумовний перехід означає передачу управління за певною адресою, не залежною від виконання логічних умов. Під час умовного переходу передача управління здійснюється за одною з двох або більше адрес залежно від виконання умов.

Для реалізації умовних переходів можна використовувати наступні мікроінструкції *CJP*, *CJV*, *JRP*, *CJPP* і *TWB*, а безумовних – *IZ*, J*MAP*, *LDCT* і *CONT* (табл. 2.27). Крім того, будь яку умовну мікроінструкцію можна перетворити на безумовну, якщо встановити на вході ВУ4 сигнал $CCE = 1$. При цьому, як вже було зазначено, інструкція виконуватиметься так, як встановиться сигнал $\alpha = 1$.

Циклічні конструкції мікропрограм зазвичай організовують з використанням мікроінструкцій *RFCT*, *RPCT*, *LOOP* і *TWB*. За цього інструкція *LOOP* забезпечує вихід з циклу за умовою, а інші – за нульовим вмістом лічильника циклів (РА/ЛЦ), тобто за кількістю повторень циклу. Для організації циклів можна також використовувати умовні переходи, вказавши в якості адреси переходу адресу мікрокоманди початку циклу.

Для переходу до мікропідпрограм зручно користуватися інструкціями *CJS*, *JSRP*, при виконанні яких адреса повернення із мікропідпрограми зберігається в стеку. Інструкція *CRTN* забезпечує повернення з мікропідпрограми за адресою, що вслід за цим виштовхується із стеку. Наявність стеку дозволяє організувати вкладені мікропідпрограми, тобто забезпечити звернення з однієї мікропідпрограми до іншої. Стек може берегти не більш ніж п'ять адрес повернення, що визначає максимальне число вкладень мікропідпрограм.

При застосуванні мнемонічного запису мікрокоманд, виконуваних схемою ФАМ можна використовувати мнемоніку найбільш розповсюджених логічних умов, що формуються СУСЗ:

– `z` – заздалегідь хибна умова (умова ніколи не виконується);
– `nz` – заздалегідь істинна умова (умова завжди виконується);
– `zo`, `co`, `no`, `vo` – входи ознак стану *IZ*, *IC*, *IN*, *IV* СУСЗ;

– `rm_z`, `rm_c`, `rm_n`, `rm_z` – розряди регістру стану *RM*: *MZ*, *MC*, *MN*, *MV*;

– `rn_z`, `rn_c`, `rn_n`, `rn_z` – розряди регістру стану *RN*: *NZ*, *NC*, *NN*, *NV*;

– `nxorv` – ознака $\overline{IN \oplus IV}$;

– `zxorc` – ознака $\overline{IC \oplus IZ}$;

– `l1`, `l2`, `l3`, `l4`, `l5`, `l6` – безпосередньо аналізувати інформацію на входах МУ: *L1*, *L2*, *L3*, *L4*, *L5*, *L6*;

– `ct`, `rdm`, `rdd`, `int`, `irq0`, `irq1`, …, `irq7` – підключати до входів МУ сигнали, що генерують різні пристрої МПС: СУСЗ, пам'ять, зовнішні пристрої, сигнали вимоги переривань.

Для підключення інвертованих логічних умов можна застосовувати ту саму мнемоніку з префіксом `not`, наприклад, `not rm_c`.

Слід зазначити, якщо умова записується з префіксом `not`, то в поле *COM* мікрокоманди встановлюється значення 1.

Як уже було зазначено формуванням значення α на вході схеми УНА на основі логічної умови (ЛУ), що надійшла на вхід МУ, управляють поля мікрокоманди *CCE* та *COM* (рис. 2.25, рис. 2.27). Далі розглянуті декілька типових комбінацій цих полів, що забезпечують формування α.

– *CCE* = 0, *COM* = 1 $\Rightarrow$ перевірка умови, α = ЛУ, наприклад, `{cjp rm_c, addr;}`;

– *CCE* = 0, *COM* = 0 $\Rightarrow$ перевірка інвертованої умови, $\alpha = \overline{\text{ЛУ}}$, наприклад, `{cjp not rm_c, addr;}`;

– *CCE* = 1, *COM* = * $\Rightarrow$ безумовний перехід, $\alpha = 1$, наприклад, `{cjp nz, addr;}`;

Реалізація мікрокоманд умовних та безумовних перехідів наведена у прикладах 2.6 – 2.11. Для виводу коду умови через шину *CT* схеми СУСЗ в розглянутих прикладах встановлюється $\overline{OECT} = 0$, що у наведених фрагментах мікрокоманд умовно не показано.

В прикладах 2.12 – 2.16 представлено застосування управляючих конструкцій БМУ, організація циклів, реалізація мікропідпрограм.

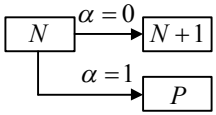
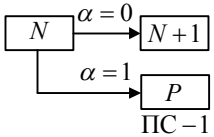
Таблиця 2.27. Управляючі конструкції БМУ

Мнемоніка мікрооперації	**($I_3..I_0$)**	**Сигнали та мікрооперації**	**Коментар**	**Графічна інтерпретація**
1	2	3	4	5
Мікрокоманди безумовних переходів				
`{jz;}` *– перехід до нульової адреси*	`0000`	A:= 0000*h* $0 \to$ показчик стека ;	ЛМК встановлюється в значення ; відбувається; очистка стека	$N \to 0$
`{cont;}` *– перехід до наступної адреси*	`1110`	A:= СМК;	Мікрокоманда призводить до інкременту ЛМК, і формування на виході *A* ФАМ адреси $A_{i+1} := A_i + 1$.	$N \to N+1$
`{jmap;}` *– безумовний перехід за адресою, що передається з Буферу M*	`0010`	$\overline{ME} = 0$; A:= *D*;	Використовується під час емуляції системи команд для переходу в ПМК до нової мікроподпрограми. При цьому на ЛШ має бути виставлена початкова адреса мікропідпрограми. Під час виконання МК формується сигнал $\overline{ME} = 0$, який відкриває Буфер *M*, і адреса нової мікропідпрограми з ЛШ надходить на вхід *D* ФАМ і далі на вихідну шину *A* (*A*:= *D*).	$N \to M$

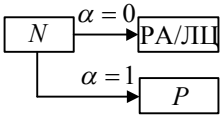
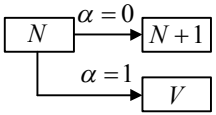
Продовження табл. 2.27.

1	2	3	4	5
`{ldct val;}` *– завантаження РА/ЛЦ та перехід до наступної адреси*	`1100`	$\overline{PE}=0$; $\overline{RLD}=0$; A:= СМК; $P \to$ РА/ЛЦ	Значення *val* у цьому випадку є вмістом поля *P* мікрокоманди. Під час виконання МК виробляється сигнал $\overline{PE}=0$, який відкриває Буфер *P*, і значення з поля *P* (кількість повторень циклу) надходить на вхід *D* ФАМ та записується в РА/ЛЦ. При цьому встановлюється $\overline{RLD}=0$.	$N \to N+1$ $P \to$ РА/ЛЦ
`{push;}` *– запис в стек*	`0100`	A:= ЛМК; ЛМК $\to$ стек ;	Мікрокоманда записує в стек наступну адресу, наприклад, якщо мікрокоманду розміщено в ПМК за адресою $2d3h$, то під час її виконання в стек буде записано значення $2d4h$.	$N \to N+1$ $N+1 \to$ стек
`{push cond, val;}` *– запис в стек та умовне завантаження РА/ЛЦ*	`0100`	$\overline{PE}=0$; A:= ЛМК; ЛМК $\to$ стек ; P $\to$ РА/ЛЦ, за $\alpha=0$ $\overline{RLD}=0$	Мікрокоманда записує в стек наступну адресу, завантажує в РА/ЛЦ значення *val*, якщо виконується умова (*cond* = 1). При цьому встановлюється $\overline{RLD}=0$. Якщо умова не виконується, то запис *val* в РА/ЛЦ не відбувається. Мікрокоманда рівноцінна двом мікрокомандам `{push;}` та `{ldct val;}`.	$N \to N+1$ $N+1 \to$ стек за $\alpha=1$, $P \to$ РА/ЛЦ

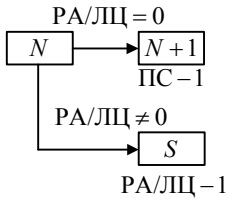
Продовження табл. 2.27.

1	2	3	4	5
		Мікрокоманди умовних переходів		
{cjp cond, addr;} **{cjp cond, ml1;}** *– умовний перехід за адресою P*	0011	$\overline{PE}=0$; за α = 0, A:= ЛМК; за α = 1, A:= D;	Мікрокоманда `cjp` (*Condition JumP*) дозволяє здійснювати розгалуження в мікропрограмах. Під час виконання мікрокоманди формується сигнал $\overline{PE}=0$, який відкриває Буфер *P*, і значення *addr* з поля *P* мікрокоманди надходить на вхід *D* ФАМ. Якщо умова виконується (*cond*=1), то *A*:= *addr*, інакше *A*:= ЛМК.	
{cjpp cond, addr;} **{cjpp cond, ml1;}** *– умовний перехід за адресою P і виштовхування із стеку*	1011	$\overline{PE}=0$; за α = 0, *A*:= ЛМК; за α = 1, *A*:= *D*, ПС=ПС – *1*;	Мікрокоманда `cjpp` (*Condition JumP and Pop*) відрізняється від МК `cjp` лише тим, що у разі виконання умови (*cond* = 1) додатково з верхівки стека виштовхується значення. Отже, цю мікрокоманду доцільно використовувати для виходу з циклу за умовою.	

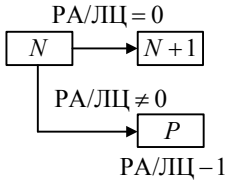
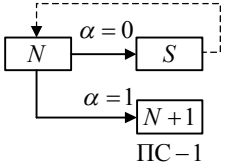
Продовження табл. 2.27.

1	2	3	4	5
`{jrp cond, addr;}` *– умовний перехід за адресою P або за адресою з РА/ЛЦ*	`1011`	$\overline{PE}=0$; за α = 0, *A*:= РА/ЛЦ; за α = 1, *A*:= *D*;	Сигнал $\overline{PE}=0$ відкриває Буфер *P*, значення *addr* з поля *P* мікрокоманди надходить на вхід *D* ФАМ. Якщо умова виконується ($cond=1$), то A:=*addr*, і відбувається перехід за адресою, вказаною в мікрокоманді; інакше *A*:=(РА/ЛЦ), і відбувається перехід за адресою, яка зберігається в РА/ЛЦ.	
`{cjv cond;}` *– умовний перехід на підпрограму обробки переривання*	`0110`	$\overline{VE}=0$; за α = 0, *A*:= ЛМК; за α = 1, *A*:= *D*;	Мікрокоманда `cjv` (*Condition Jump by Vector*) дозволяє здійснити перехід на мікропрограму обслуговування переривання (драйвер). На Буфер *V* надходять сигнали від зовнішніх пристроїв системи. Під час виконання цієї мікрокоманди формується сигнал $\overline{VE}=0$, який відкриває Буфер *V*, і початкова адреса драйвера надходить на вхід *D* ФАМ. Якщо умова виконується ($cond=1$), то A:= *V*, інакше – A:= ЛМК.	

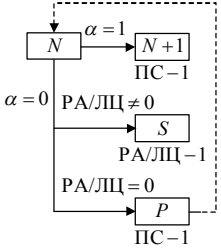
Продовження табл. 2.27.

1	2	3	4	5
Мікрокоманди для організації циклів				
`{rfct;}` *– повторити цикл (перехід за адресою з верхівки стека), якщо РА/ЛЦ ≠0*	`1000`	$\overline{PE} = 0$; за РА/ЛЦ ≠ 0, *A*:= Стек, РА/ЛЦ:=РА/ЛЦ – 1; за РА/ЛЦ = 0, *A*:=ЛМК, ПС:=ПС–1;	Використовують для багаторазового повторення певної частини мікропрограми, якщо відома кількість повторень. Кількість повторень перед початком виконання циклу записують в РА/ЛЦ за допомогою мікрокоманди `{push cond, val;}`. При цьому в стек заноситься адреса початку циклу. Після виконання тіла циклу, який слід завершити мікрокомандою `{rfct;}`, аналізується вміст РА/ЛЦ. Якщо РА/ЛЦ ≠ 0, здійснюється перехід на початок циклу (за адресою зі стека), а значення в РА/ЛЦ зменшується на 1. Повторення циклу відбувається, поки виконується РА/ЛЦ ≠ 0. Інакше (РА/ЛЦ = 0) здійснюється перехід до наступної мікрокоманди й виштовхування зі стека. Механізм виконання мікрокоманди `{rfct;}`такий, що декримент РА/ЛЦ відбувається після аналізу вмісту РА/ЛЦ. Тому завжди тіло циклу буде виконане на один раз більш ніж указано у параметрі `val`, вслід чого необхідна кількість циклів має бути зменшена на одиницю.	

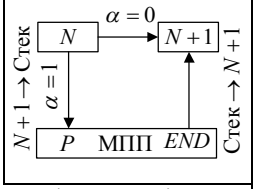
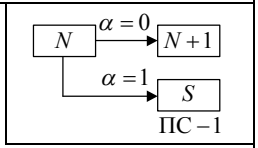
Продовження табл. 2.27.

1	2	3	4	5
{rpct addr;} *– повторити цикл (перехід за адресою P), якщо РА/ЛЦ = 0*	`1001`	$\overline{PE} = 0$; за РА/ЛЦ ≠ 0, *A*:= *D*, РА/ЛЦ:=РА/ЛЦ – 1; за РА/ЛЦ = 0, *A*:= ЛМК;	У мікрокоманді перехід відбувається за адресою *addr* з поля *P* мікрокоманди, що надійшла на вхід *D* ФАМ через Буфер *P* ($\overline{PE} = 0$). У іншому механізм роботи МК співпадає із МК `{rfct;}`, що розглянута раніш.	
{loop cond;} *– вихід з циклу за умовою*	`1101`	$\overline{PE} = 0$; за α = 0, *A*:= *S*; за α = 1, *A*:= ЛМК, ПС:=ПС – 1;	Мікрокоманда `{loop cond;}` використовується для завершення тіла циклу, якщо цикл має заздалегідь невідому кількість повторень. Якщо умова $cond = 0$, то здійснюється перехід на початок циклу (за адресою збереженою у верхівці стеку), інакше здійснюється перехід до наступної мікрокоманди, що розташована наступною після МК завершення циклу і виштовхування зі стека адреси початку циклу.	

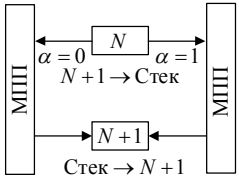
Продовження табл. 2.27.

1	2	3	4	5
`{twb cond, addr;}` *– розгалуження на три напрями*	`1111`	$\overline{PE} = 0$; за (α = 0) & (РА/ЛЦ ≠ 0), *A*:= Стек, РА/ЛЦ:=РА/ЛЦ – 1; за (α = 0) & (РА/ЛЦ = 0), *A*:= *D*, ПС:=ПС–*1;* за α = 1, *A*:= ЛМК, ПС:=ПС–*1;*	Під час виконання мікрокоманди `{twb cond, addr;}` аналізуються дві умови: *cond*, що явно вказана в мікрокоманді, та вміст РА/ЛЦ, що порівнюється з нулем. Якщо виконуються дві умови РА/ЛЦ≠0 і $cond = 0$, то здійснюється перехід за адресою з верхівки стека А:= Стек і вміст РА/ЛЦ зменшується на одиницю. Якщо РА/ЛЦ=0 і $cond = 0$, то здійснюється перехід за адресою *addr* з поля *P* мікрокоманди, що надійшла на вхід *D* ФАМ через Буфер *P* ($\overline{PE} = 0$), і виштовхування адреси повернення на початок циклу зі стека. Якщо виконується умова $cond = 1$, то здійснюється перехід до наступної після виконання циклу адреси А:=ЛМК.	

Продовження табл. 2.27.

1	2	3	4	5
Мікрокоманди для роботи з мікропідпрограмами				
`{cjs cond, addr;}` *– виклик мікропрограми за умовою*	`0001`	$\overline{PE}=0$; за α = 0, *A*:= ЛМК; за α = 1, *A*:= *D*, ЛМК → Стек ;	Мікрокоманда `{cjs cond, addr;}` (*Condition Jump to Subroutine*) забезпечує реалізацію механізму виклику мікропідпрограми за умовою, відбувається перехід на мікропідпрограму, що знаходиться в ПМК за адресою *addr*, якщо вказана умова виконується ($cond=1$). При цьому в стек записується адреса повернення (тобто адреса наступної за `cjs` мікрокоманди), а на вихідну шину *A* ФАМ надходить *addr* (з поля *P* через Буфер *P*), інакше здійснюється перехід до наступної мікрокоманди.	
`{crtn cond;}` *– умовне повернення з мікропідпрограми*	`1010`	$\overline{PE}=0$; за α = 0, *A*:= ЛМК; за α = 1, *A*:= Стек, ПС:=ПС–*1;*	Мікрокоманда `{crtn cond;}` (*Condition ReTurN*) забезпечує повернення з мікропідпрограми. Якщо виконується умова $cond=1$, то виконання мікропідпрограми припиняється, здійснюється повернення в основну мікропрограму за адресою з верхівки стеку (*A*:= Стек) і виштовхування зі стека. Інакше відбувається перехід до наступної мікрокоманди мікропідпрограми.	

Продовження табл. 2.27.

1	2	3	4	5
`{jsrp cond,addr;}` *– перехід до однієї з двох мікропідпрограм*	`0101`	$\overline{PE}=0$; за α = 0, *A*:= РА/ЛЦ, ЛМК → Стек ; за α = 1, *A*:= *D;*	Під час виконання мікрокоманди `{jsrp cond,addr;}` в стек записується адреса повернення з МПП (тобто адреса наступної за `JSRP` мікрокоманди) та здійснюється виклик однієї з двох мікропідпрограм – мікропідпрограми, що розміщена за адресою *addr*, якщо виконується умова $cond=1$. (за цього значення *addr* з поля *P* МК через Буфер *P* надходить на вихід *A* ФАМ, під час встановлення сигналу $\overline{PE}=0$), або мікропідпрограми, що розміщена за адресою з РА/ЛЦ (за цього А:=РА/ЛЦ).	

Примітки: *N* – адреса виконуваної МК; *cond* – логічна умова; *addr* – адреса переходу; інші позначення відповідають рис. 2.27.

Приклад 2.6. Якщо результат обчислення суми у поточному такті від'ємний ($NO = 1$), перейти за адресою 0007*H* (за міткою `m1`), при цьому виконати виштовхування даних зі стеку, інакше продовжити обчислення.

Мікрокоманда

Адреса [16]	*D* [16]	БМУ_МІ [2]	*CCE*	*COM*	*CI*	$\overline{RLD}$	*MS*	СУСЗ_МІ $I_5 - I_0$	*P*	*CT*
0000	0007	1011	0	1	1	1	1	111110	0007	*IN*

Мнемонічний запис

```
          link l1:ct;
000       {add r1,r1,r0;cjpp no, m1;}
...       {}
007  m1   {}
```

Приклад 2.7. Виконати безумовний перехід на мітку `m1`, інакше перейти за адресою з РА/ЛЦ.

Мікрокоманда

Адреса [16]	*D* [16]	БМУ_МІ [2]	*CCE*	*COM*	*CI*	$\overline{RLD}$	*MS*	СУСЗ_МІ $I_5 - I_0$	РА/ЛЦ	*P*	*CT*
0000	0007	0111	1	*	1	1	*	000000	000*A*	0007	*

Мнемонічний запис

```
0000      {jrp nz,m1;}
          {}
0007 m1   {}
          ...
000A      {}
```

Приклад 2.8. Якщо результат виконання підсумовування не дорівнює нулю то перейти на мікроподпрограму за адресою, що надійшла з Буферу *V*.

Мікрокоманда

Адреса [16]	*D* [16]	БМУ_МІ [2]	*CCE*	*COM*	*CI*	$\overline{RLD}$	*MS*	СУСЗ_МІ $I_5 - I_0$	*V*	*CT*
0000	0007	0110	0	0	1	1	2	110100	0007	*IZ*

Мнемонічний запис

```
      link l2:ct;
0000  {add r1,r1,z;cjv not zo;}
```

Приклад 2.9. Завантажити у РА/ЛЦ значення, якщо виконується умова CT = 1.

Мікрокоманда

Адреса [16]	*D* [16]	БМУ_МІ [2]	*CCE*	*COM*	*CI*	$\overline{RLD}$	*MS*	СУСЗ_МІ $I_5 - I_0$	*P*	*CT*
0000	0006	0100	1	0	1	0	1	000100	0006	*NZ*

Мнемонічний запис

```
      link l1:ct;
0000  {push ct,6;}          \ якщо CT=1, то РА/ЛЦ:=6
```

Приклад 2.10. Якщо отримано нульовий результат (ZO = 1), то перейти на мітку `ml1`, інакше продовжити обчислення.

Мікрокоманда

Адреса [16]	*D* [16]	БМУ_МІ [2]	*CCE*	*COM*	*CI*	$\overline{RLD}$	*MS*	СУСЗ_МІ $I_5 - I_0$	*P*	*CT*
0000	0007	0011	0	1	1	1	1	110100	0007	*IZ*

Мнемонічний запис

```
           link l1:ct;
0000       {cjp zo,ml1;}
           {}
0007  ml1  {}
```

Приклад 2.11. Якщо $NV = 1$, то здійснюється виклик мікроподпрограми за адресою *SUB*.

Мікрокоманда

Адреса [16]	D [16]	БМУ_МІ [2]	CCE	COM	CI	$\overline{RLD}$	MS	СУСЗ_МІ $I_5 - I_0$	P	CT
0000	0007	0001	0	1	1	1	4	000110	0007	NV

Мнемонічний запис

```
        link l4:ct;
0000    {cjs rn_v, sub;}      \ перевірка логічної умови
                              \ якщо NV = 1 – перехід на МПП
         ...
0007 sub {}                   \ початок МПП SUB
```

Приклад 2.12. Організувати цикл із заданою кількістю повторень можна в такий спосіб:

Мікрокоманда

Адреса [16]	*D* [16]	БМУ_МІ [2]	*CCE*	*COM*	*CI*	$\overline{RLD}$	$\overline{OECT}$	*MS*	СУСЗ_МІ $I_5 - I_0$	Коментарі		
										P	Стек	*CT*
0000	000*C*	0100	1	*	1	0	1	*	000000	000*C*	0001	*
0001	****	1110	*	*	1	1	1	*	000000	****	0001	*
. . .												*
0009	000*F*	1011	0	1	1	1	0	1	101010	000*F*	0000	*MC*
. . .												
000E	****	1000	*	*	1	1	1	*	000000	****	0001	*
000F	****	1110	*	*	1	1	1	*	000000	****	****	*

Мнемонічний запис

```
          link l1:ct;
0000      {push nz,12;}        \ кількість повторень циклу
0001      {}                   \ початок циклу
           ...
0009      {cjpp rm_c,mex}      \ перевірка логічної умови, якщо МС = 1 – вихід з циклу
           ...
000E      {rfct;}              \ кінець циклу
000F   mex{}                   \ продовження мікропрограми
```

Приклад 2.13. Організувати цикл із виходом за умовою можна у такий спосіб:

Мікрокоманда

Адреса [16]	*D* [16]	БМУ_МІ [2]	*CCE*	*COM*	*CI*	$\overline{RLD}$	$\overline{OECT}$	*MS*	СУСЗ_МІ $I_5 - I_0$	Стек	*CT*
0000	****	0100	0	*	1	1	1	*	000000	0001	*
0001	****	1110	*	*	1	1	1	*	000000	0001	*
. . .											
000E	****	1101	0	0	1	1	0	3	101111	0001	*MN*
000F	****	1110	*	*	1	1	1	*	000000	****	*

Мнемонічний запис

```
        link l3:ct;
0000    {push;}                    \ завантаження в стек адреси початку циклу
0001    {}                         \ початок циклу
         ...
000E    {loop not rm_n;}           \ перевірка логічної умови: якщо MN = 0 – вихід з циклу,
                                   \  інакше – повернення на адресу із стеку
000F    {}                         \ продовження мікропрограми
```

Приклад 2.14. Розробити мікропрограму для обчислення заданої функції:

$$D = C/2 + 2A(B+1)$$

Виконання завдання:

Алгоритм обчислення функції наведений на рис. 2.29. Цифрова діаграма стану регістрів наведена у табл. 2.28. Мікропрограма налагоджена у моделюючому комплексі COMPLEX зображена на рис. 2.30.

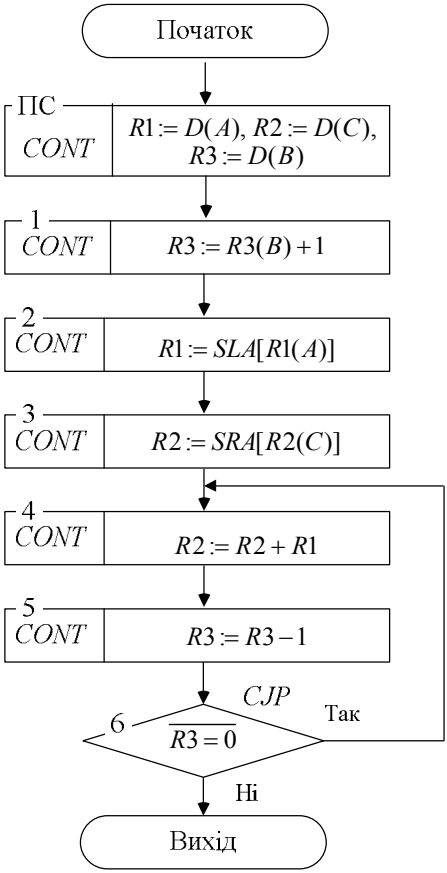


Рис. 2.29. Алгоритм обчислення функції

Таблиця 2.28. Цифрова діаграма стану регістрів

№ такту	***R*1 (*A*)**	***R*2(*C*)**	***R*3(*B*)**	***ZO***	**Коментарі**
ПС	0005	000C	0005	0	Завантаження даних
1			0006		$R3 := R3 + 1$ $(B + 1)$
2	000A				$R2 := l[R2].0$ $(2A)$
3		0006			$R1 := 0.r[R1]$ $(C/2)$
4(1) 5(1) 6(1)		0010	0005	0	$R2 := R2 + R1$ $(A+C)$ $R3 := R3 - 1$ $(B - 1)$ $ZO = 0$ $(B \neq 0)$
4(2) 5(2) 6(2)		001A	0004	0	$R2 := R2 + R1$ $(A+C)$ $R3 := R3 - 1$ $(B - 1)$ $ZO = 0$ $(B \neq 0)$
4(3) 5(3) 6(3)		0024	0003	0	$R2 := R2 + R1$ $(A+C)$ $R3 := R3 - 1$ $(B - 1)$ $ZO = 0$ $(B \neq 0)$
4(4) 5(4) 6(4)		002E	0002	0	$R2 := R2 + R1$ $(A+C)$ $R3 := R3 - 1$ $(B - 1)$ $ZO = 0$ $(B \neq 0)$
4(5) 5(5) 6(5)		0038	0001	0	$R2 := R2 + R1$ $(A+C)$ $R3 := R3 - 1$ $(B - 1)$ $ZO = 0$ $(B \neq 0)$
4(6) 5(6) 6(6)		0042	0000	1	$R2 := R2 + R1$ $(A+C)$ $R3 := R3 - 1$ $(B - 1)$ $ZO = 1$ $(B = 0)$
7					Кінець обчислення

COMPLEX

Файл Редактор Выполнить Протокол Помощь Опции Выход

Исходный файл :FUNC_1.PMK Страница: 1.1

Адр	Конст. БОД БМУ	Конст. БОД OED	Б О Д В С 1 MI*	A	B	OEY	В Р 2 MI*	C	Z	N	V	OECT	CEN	CEM	SE
000	0005	0	011.011.101	1	1	1	00.00000.000000	0	0	0	0	1	1	1	1
001	000C	0	011.011.101	2	2	1	00.00000.000000	0	0	0	0	1	1	1	1
002	0005	0	011.011.101	3	3	1	00.00000.000000	0	0	0	0	1	1	1	1
003	0000	1	011.000.100	3	3	1	01.00000.000000	0	0	0	0	1	1	1	1
004	0000	1	111.000.100	1	1	1	00.10000.000000	0	0	0	0	1	1	1	0
005	0000	1	101.000.100	2	2	1	00.00010.000000	0	0	0	0	1	1	1	0
006	0000	1	011.000.001	1	2	1	00.00000.000000	0	0	0	0	1	1	1	1
007	0000	1	011.001.100	3	3	1	00.00000.000000	0	0	0	0	1	1	1	1
008	0006	1	011.000.100	3	3	1	00.00000.110100	0	0	0	0	0	1	1	1
009	0000	1	001.000.000	0	0	1	00.00000.000000	0	0	0	0	1	1	1	1

Исходный файл :FUNC_1.PMK Страница: 1.2

Адр	РАА,РАВ MSA	MSB	EWA	EWB	БМУ и БПП ВУ4 MI*	CCE	COM	CI	RLD	MS	ВН1 MI*	EINS	ПАВ EV	ОП,ВУ,РА ВУ I	O	LCK	IA	ОП R	W	РгА EWH	EWL
000	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
001	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
002	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
003	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
004	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
005	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
006	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
007	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
008	0	0	1	1	0011	0	0	1	1	1	0000	1	1	1	1	1	1	1	1	1	1
009	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1

Нажмите любую клавишу для окончания

Исходный файл :FUNC_1_A.ASM Страница: 1

```
/------connection area------
link l1:ct
/------swap of registers area------
accept R1:5h
accept R2:0Ch
accept R3:5h
/------programs area------
        {add R3,R3,Z,NZ;}
        {add SLL,R1,R1,Z;}
        {add SRL,R2,R2,Z;}
l11     {add R2,R1,R2;}
        {sub R3,R3,Z,Z;}
        {add R3,R3,Z;cjp not Z0,l11;}
        {}
```

Рис. 2.30. Мікропрограма обчислення функції

Приклад 2.15. Розробити мікропрограму для блоку обробки даних, що реалізує заданий мікроалгоритм (рис. 2.31).

Виконання завдання:

На вихідному мікроалгоритмі (рис. 2.31) біля кожної вершини указані адреси розміщення відповідних мікрокоманд у ПМК, а також указані виконувані мікроінструкції для ІС К1804ВУ4, що забезпечують формування адрес мікрокоманд. Адреса першої вершини може бути довільною. Кодова карта з відповідними коментарями наведена у табл. 2.29. Кожній вершині мікроалгоритму відповідає рядок табл. 2.9, номер якої співпадає з номером вершини. Адреси та дані подані у шістнадцятирічній системі числення. Розряди мікрокоманди, значення яких можуть буди довільними, позначені символом «*».

У стовпці *LC* показані логічні умови, що впливають на формування адреси наступної мікрокоманди. Під час налагодження мікропрограми слід перевіряти правильність переходів, як за нульовим значенні умови, так і за одиничним.

На кодовій карті умовно не показані поля мікрокоманди, що не змінюються під час виконання мікропрограми.

Даний приклад ілюструє виконання кожної з шістнадцяти мікроінсторкції, що реалізує БМУ.

Умовні інструкції у вершинах 4, 5, 6, 8, 10, 15, 24 мікроалгоритму відповідають безумовному переходу. Безумовний перехід забезпечується поданням одиничного сигналу на вхід $CCE = 1$ (табл. 2.29). Слід зазначити, що безумовний перехід можна реалізувати також шляхом підключення до входу СС виходу *L*7 мультиплексора, що відповідає логічній одиниці. За цього у полях мікрокоманди треба встановити наступні сигнали: MS = 7[16] – номер виходу мультиплексора, $CCE = 0$, $COM = 1$ (або $COM = 0$, якщо умова має бути інвертована) та $\overline{OECT} = 0$. Такий спосіб застосований під час кодування мікрокоманд на рис. 2.32.

Мікропрограма у машинних кодах та із застосуванням символічного мікроасемлеру, налагоджена у моделюючому комплексі *COMPLEX*, зображена на рис. 2.32, *а*, *б* відповідно.

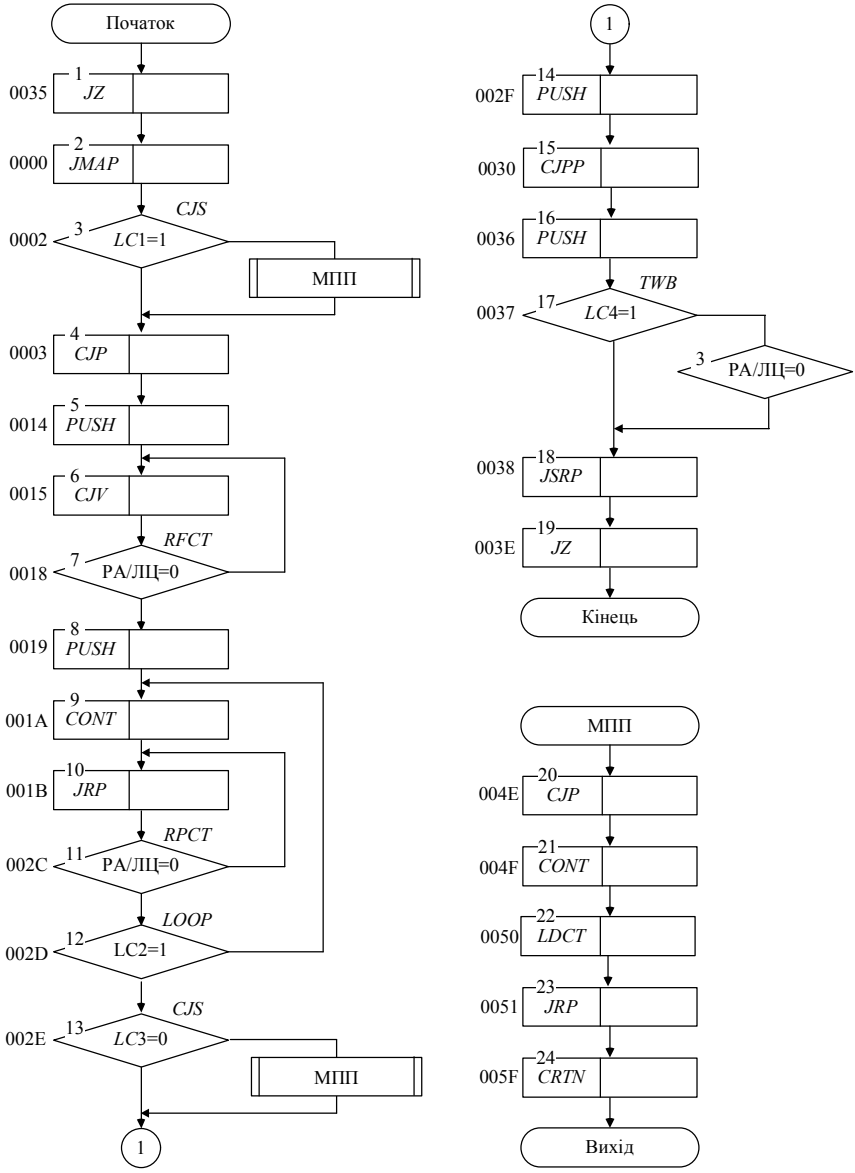


Рис. 2.31. Вихідний мікроалгоритм

Таблиця 2.29. Кодова карта

Мікрокоманда											**Коментарі**		
№ пп.	**Мнє-моніка**	**Ад-реса** [16]	**Константа БОД**		**ВУ4 (БМУ)**								
			D [16]	$\overline{OED}$	БМУ_МІ [2]	*CCE*	*COM*	*CI*	$\overline{RLD}$	*MS*	*P* (*M*,*V*)	*ПС: СТЕК*	*LC*
1	*JS*	035	0000	1	0000	*	*	1	1	0	*	0:*	*
2	*JMAP*	000	0000	1	0010	*	*	1	1	*	0002(*M*)		*
3	*CJS*	002	004E	1	0001	0	0	1	1	1	004E	1:003	$\overline{L1=0}$
4	*CJP*	003	0014	1	0011	1	1	1	1	*	0014		1(*NZ*)
5	*PUSH*	014	0003	1	0100	1	1	1	0	*	0003	1:015	1(*NZ*)
6	*CJV*	015	0000	1	0110	1	1	1	1	*	0018(*V*)		1(*NZ*)
7	*RFCT*	018	0000	1	1000	*	1	1	1	*	*	0:*	РА/ЛЦ=0
8	*PUSH*	019	0002	1	0100	1	1	1	1	*	0002	1:01A	1(*nz*)
9	*CONT*	01A	0000	1	1110	*	1	1	1	*	*		*
10	*JRP*	01B	002C	1	0111	1	1	1	1	*	002C		1(*NZ*)
11	*RPCT*	02C	001B	1	1001	*	1	1	1	*	001B		РА/ЛЦ=0
12	*LOOP*	02D	0000	1	1101	0	1	1	1	2	*	0:*	*L*2=1

Продовження таблиці 2.29.

Мікрокоманда											**Коментарі**		
№ пп.	**Мнє-моніка**	**Ад-реса** [16]	**Константа БОД**		**ВУ4 (БМУ)**								
			D [16]	$\overline{OED}$	БМУ_МІ [2]	*CCE*	*COM*	*CI*	$\overline{RLD}$	*MS*	*P (M,V)*	*ПС: СТЕК*	*LC*
13	*CJS*	02E	04E	1	0001	0	0	1	1	3	04E	1:02F	$\overline{L3=0}$
14	*PUSH*	02F	0000	1	0100	*	1	1	0	*	0000	1:030	*
15	*CJPP*	030	0036	1	1011	1	1	1	1	*	0036	0:*	1(*NZ*)
16	*PUSH*	036	0005	1	0100	0	1	1	0	7	0005	1:037	*L*7
17	*TWB*	037	003E	1	1111	0	0	1	1	4	003E	0:*	$\overline{L4=0}$
18	*JSRP*	038	003E	1	0101	0	1	1	1	7	003E	1:039	*L*7
19	*JZ*	038	0000	1	0000	*	*	1	1	0	*	0:*	*
20	*CJP*	04E	005F	1	0011	0	0	1	1	7	005F		$\overline{1(NZ)}$
21	*CONT*	04F	0000	1	1110	0	1	1	1	*	*		*
22	*LDCT*	050	005F	1	1100	1	1	1	0	*	005F		*
23	*JRP*	051	004F	1	0111	0	1	1	1	0	004F		0(*Z*)
24	*CRTN*	05F	0000	1	0111	1	1	1	1	*	*	0:*	1(*NZ*)

Адр	Конст. БОД		Б О Д В С 1					БМУ ВУ4					
	БМУ	OED	MI *	A	B	OEY	OECT	MI *	CCE	COM	CI	RLD	MS
035	0000	1	001.000.000	0	0	1	1	0000	0	1	1	1	0
000	0000	1	001.011.100	1	0	0	1	0010	0	1	1	1	0
001	0000	1	001.000.000	0	0	1	1	1110	0	1	1	1	0
002	004E	1	001.000.000	0	0	1	1	0001	0	0	1	1	1
003	0014	1	001.000.000	0	0	1	0	0011	0	1	1	1	7
014	0003	0	001.000.000	0	0	1	0	0100	0	1	1	1	7
015	0000	1	001.000.000	0	0	1	0	0110	0	1	1	1	7
018	0000	1	001.000.000	0	0	1	1	1000	0	1	1	1	0
019	0002	0	001.000.000	0	0	1	0	0100	0	1	1	1	7
01A	0000	1	001.000.000	0	0	1	1	1110	0	1	1	1	0
01B	002C	1	001.000.000	0	0	1	0	0111	0	1	1	1	7
02C	001B	1	001.000.000	0	0	1	1	1001	0	1	1	1	0
02D	0000	1	001.000.000	0	0	1	1	1101	0	1	1	1	2
02E	004E	1	001.000.000	0	0	1	1	0001	0	0	1	1	3
02F	0000	1	001.000.000	0	0	1	1	0100	0	1	1	1	0
030	0036	1	001.000.000	0	0	1	0	1011	0	1	1	1	7
036	0005	0	001.000.000	0	0	1	0	0100	0	1	1	1	7
037	003E	1	001.000.000	0	0	1	1	1111	0	0	1	1	4
038	003E	1	001.000.000	0	0	1	0	0101	0	1	1	1	7
03E	0000	1	001.000.000	0	0	1	1	0000	0	1	1	1	0
04E	005F	1	001.000.000	0	0	1	0	0011	0	0	1	1	7
04F	0000	1	001.000.000	0	0	1	1	1110	0	1	1	1	0
050	005F	1	001.000.000	0	0	1	1	1100	0	1	1	1	0
051	0000	1	001.000.000	0	0	1	0	0111	0	1	1	1	0
05F	0000	1	001.000.000	0	0	1	0	1010	0	1	1	1	7

Рис. 2.32. Результати моделювання: а –мікропрограма в машинних кодах, б – мікропрограма у мнємокодах.

```
Исходный файл :PROG1.ASM  Страница: 1

        link m:12,11,10,9,8,7,6,5,4,3,2,1
        link v:z,z,z,z,z,z,z,nz,nz,z,z,z
        link l2:nz
        accept r1:0005
        org 0035h
        {jz;}
        org 0000h
        {or nil,r1,z;oey;jmap;}
        {cont;}
        {cjs not l1,mpp1;}
        {cjp nz, l11;}
        org 0014h
l11     {push nz,0003;}
        {cjv nz;}
        org 0018h
l12     {rfct;}
        {push nz,0002;}
        {cont;}
l14     {jrp nz,l13;}
        org 002ch
l13     {rpct l14;}
        {loop l2;}
        {cjs not l3,mpp1;}
        {push;}
        {cjpp nz,l15;}
        org 0036h
l15     {push nz,0005h;}
        {twb not l4,003eh;}
        {jsrp nz,l16;}
        org 003eh
l16     {jz;}
        org 004eh
mpp1    {cjp not nz,l17;}
        {cont;}
        {ldct 005fh;}
        {jrp z,0000;}
        org 005fh
l17     {crtn nz;}
```

Продовження рисунку 2.32

Приклад 2.16. Розробити мікропрограму для блоку обробки даних, що реалізує заданий мікроалгоритм (рис. 2.33).

Виконання завдання:

Мікропрограма у машинних кодах, налагоджена у моделюючому комплексі *COMPLEX*, зображена на рис. 2.34.

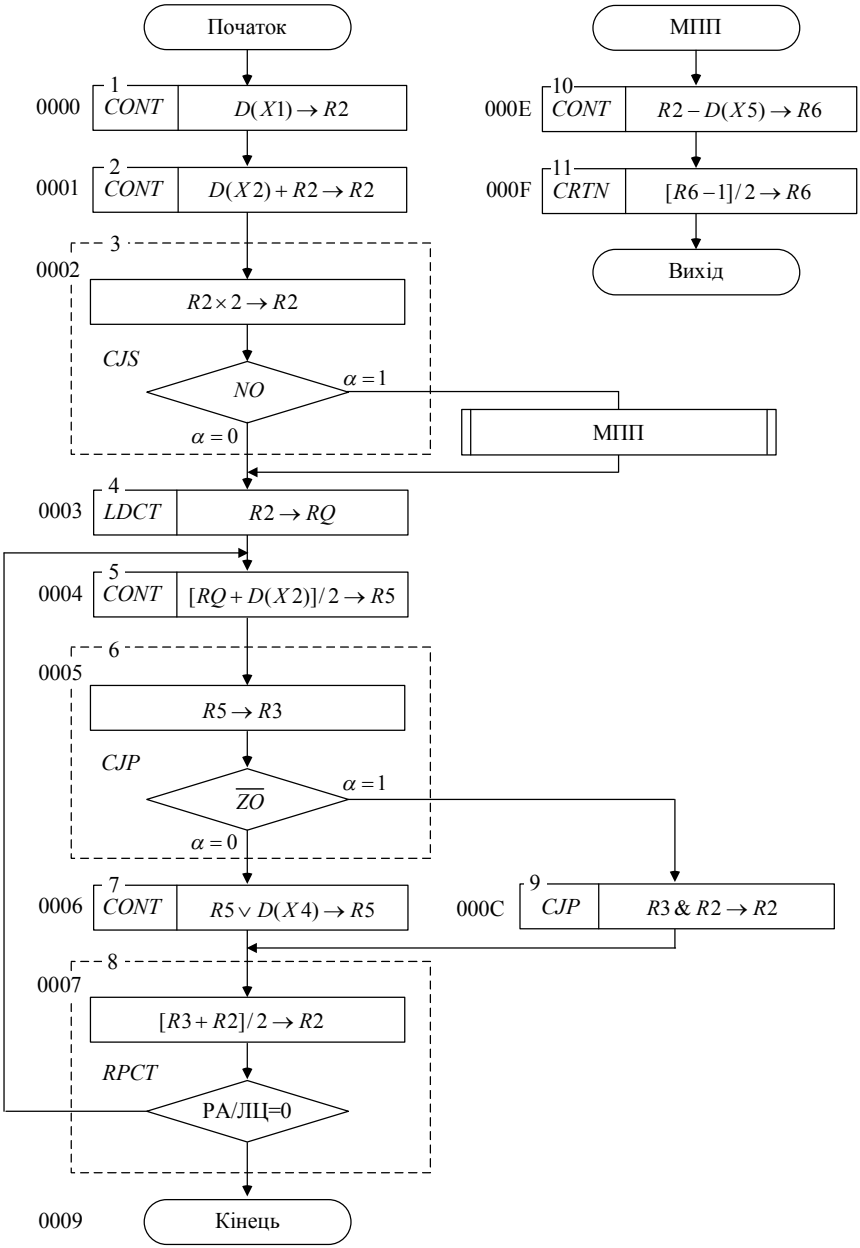


Рис. 2.33. Вихідний мікроалгоритм

```
     / мікропрограма із застосуванням символічного мікроасемблеру
     / область налагодження зв'язків
          link l2:ct;
     / область програми
0000      {or r2,r2,005h;}                    / X1(005h) → R2
0001      {add r2,r2,0c0h,z;}                 / X2(0C0h) + R2 → R2
0002 ml1  {or sl.16 r2,r2,z;cjs no,mpp;}      / R2×2 → R2 ; перехід на мікропідпрограму,
                                              / якщо результат від'ємний
0003 lab3 {add rq,r2,z;ldct 6;}               / R2 → RQ ; завантаження в РА/ЛЦ кількості
                                              / циклів
0004      {add sr.2 r5,rq,055h,z;}            / (RQ + X3(0C0h))/2 → R5
0005      {add r3,r5,z,z;cjp not zo,lab1;}    / пересилання R5 → R3 ; умовний перехід на
                                              / мітку, якщо результат не дорівнює нулю
0006      {or r5,r5,0f00fh;}                  / R5 ∨ X4(0F00Fh) → R5
0007 lab2 {add sr.2 r2,r3,r2,z;rpct,lab3;}    /[R3 + R2]/2 → R2 ; інкремент вмісту РА/ЛЦ
                                              / та перевірка, якщо РА/ЛЦ не дорівнює нулю,
                                              / повернення на початок циклу
0009      {cjp nz, lend}                      / безумовний перехід на кінець мікропрограми
000C lab1 {and r2,r2,r3;cjp nz,lab2;}         / R3 & R2 → R2 , безумовний перехід на мітку
000E mpp  {sub r6,r2,0013h}                   / R2 − X5(0013h) → R6
000F      {sub r6,r6,z,nz;crtn;}              /[R6 − 1]/2 → R6 , безумовний вихід із
                                              / мікропідпрограми
0010 lend{}                                   / кінець
```

COMPLEX

Файл Редактор Выполнить Протокол Помощь Опции Выход

Исходный файл :VVAVA.PMK Страница: 1.1

Адр	Конст. БОД БМУ	OED	Б О Д ВС1 МІ*	A	B	OEY	ВР2 МІ*	C	Z	N	V	OECT	CEN	CEM	SE
000	0005	0	011.000.111	0	2	1	00.00000.000000	0	0	0	0	1	1	1	1
001	02C0	0	101.000.101	2	2	1	00.00000.000000	0	0	0	0	1	1	1	1
002	000E	0	111.000.001	2	2	1	00.10000.111110	1	0	0	0	0	1	0	0
003	0006	0	000.000.100	2	0	1	00.00000.000000	0	0	0	0	1	1	1	1
004	0055	0	101.000.110	0	5	1	00.00010.000000	0	0	0	0	1	1	1	0
005	000C	0	011.000.100	5	3	1	00.00000.110101	0	0	0	0	1	1	1	1
006	F00F	0	011.011.101	5	5	1	00.00000.000000	0	0	0	0	1	1	1	1
007	0000	1	101.000.001	3	2	1	00.00010.000000	0	0	0	0	1	1	1	0
008	0004	1	001.000.011	0	0	1	00.00000.000000	0	0	0	0	1	1	1	1
009	0000	1	001.000.000	0	0	1	00.00000.000000	0	0	0	0	1	1	1	1
00A	0000	1	001.000.000	0	0	1	00.00000.000000	0	0	0	0	1	1	1	1
00B	0000	1	001.000.000	0	0	1	00.00000.000000	0	0	0	0	1	1	1	1
00C	0007	0	011.100.000	5	5	1	00.00000.000000	0	0	0	0	1	1	1	1
00D	0000	1	001.000.000	0	0	1	00.00000.000000	0	0	0	0	1	1	1	1
00E	0013	0	011.001.101	2	6	1	01.00000.000000	0	0	0	0	1	1	1	1
00F	0000	1	101.001.000	0	0	1	00.00010.000000	0	0	0	0	1	1	1	1
010	0000	1	001.000.000	0	0	1	00.00000.000000	0	0	0	0	1	1	1	1

Исходный файл :VVAVA.PMK Страница: 1.2

Адр	РАА,РАВ MSA	MSB	EWA	EWB	БМУ и БПП ВУ4 МІ*	CCE	COM	CI	RLD	MS	ВН1 МІ*	EINS	ПАВ EV	ОП,ВУ,РА ВУ I	O	LCK	IA	ОП R	W	РгА EWH	EWL
000	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
001	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
002	0	0	1	1	0001	0	0	1	1	2	0000	1	1	1	1	1	1	1	1	1	1
003	0	0	1	1	1100	1	1	1	0	0	0000	1	1	1	1	1	1	1	1	1	1
004	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
005	0	0	1	1	0011	0	0	1	1	2	0000	1	1	1	1	1	1	1	1	1	1
006	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
007	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
008	0	0	1	1	1001	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
009	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
00A	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
00B	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
00C	0	0	1	1	0011	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
00D	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
00E	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
00F	0	0	1	1	1010	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1
010	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	1

Нажмите любую клавишу для окончания

Рис. 2.34. Мікропрограма в машинних кодах