2.7. Проектування мікропроцесорних систем на секціонованому комплекті інтегральних схем

Комплект ІС серії К1804 дозволяє проектувати МПС, параметри яких можуть змінюватися у широких межах і мати довільну систему команд. Для побудови найпростішого процесора потрібні ІС трьох типів: К1804ВС1, К1804ВУ4 і ПЗП (наприклад, К556РТ5). Для побудови *n*-розрядного процесора потрібно $n/4$ процесорних елементів К1804ВС1. Блок мікропограмного управління можна побудувати на одній ІС К1804ВУ4 і декількох ІС К556РТ5, що є постійними запам'ятовуючими пристроями об'ємом 512×8 біт, число ПЗП визначається довжиною слова мікрокоманди.

Параметри МПС багато в чому залежать від способу організації зв'язків між управляючою частиною та операційною частиною (ОЧ) процесора. Основними структурними елементами процесора є схема формування адреси мікрокоманд К1804ВУ4, пам'ять мікрокоманд (ПМК) і процесорні елементи К1804ВС1, з яких складається ОЧ процесора.

Розглянемо декілька різних структур процесорів, які відображають найбільш істотні зв'язки, що впливають на побудову мікропрограм і параметри процесора.

У процесорі на рис. 2.38, *а* застосовується регістр адреси мікрокоманди (РАМК). В цьому випадку виконання такту починається із завантаження адреси *A* поточної мікрокоманди в РАМК за позитивним перепадом синхросигналу *CLK*. Відповідно до адреси *A* з ПМК вибирається мікрокоманда *I*(*A*), в результаті виконання якої в ОЧ формуються ознаки *S*(*A*). Ці ознаки через мультиплексор МУ можуть поступати на вхід умови *CC* схеми ФАМ та враховуватися при формуванні наступної адреси $(A+1)$ наступної мікрокоманди. Найбільш тривалий шлях проходження сигналів показаний на рис. 2.38 пунктирною лінією. Максимальна тривалість такту *T*, як випливає з діаграми на рис.2.37, дорівнює сумарній затримці сигналів у всіх блоках. Суміщати в часі виконання однієї мікрокоманди з вибіркою із ПМК іншої в даному випадку є неможливим. Такий спосіб виконання мікрокоманд називають послідовним.

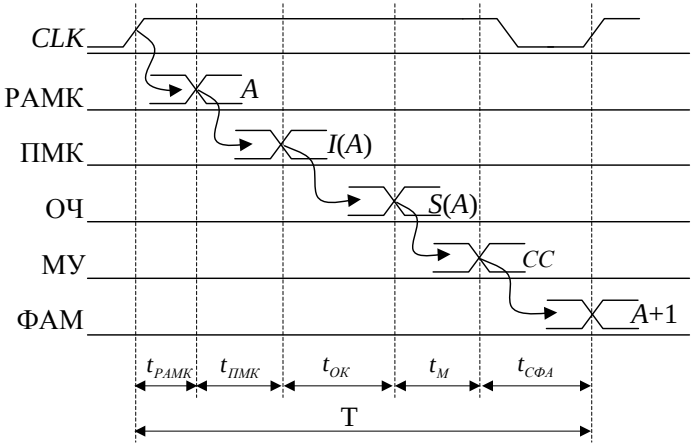


Рис. 2.37. Часова діаграма

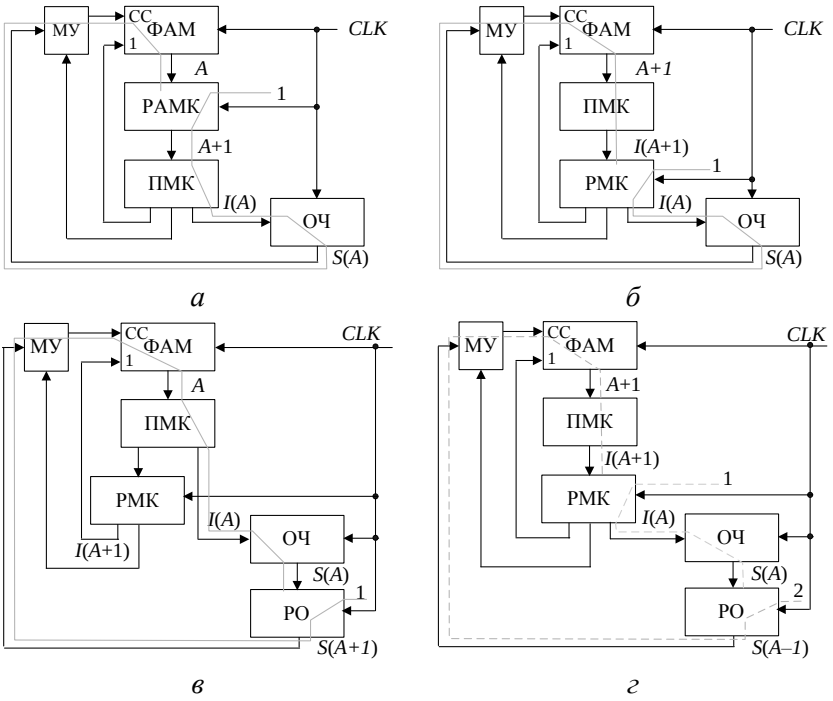
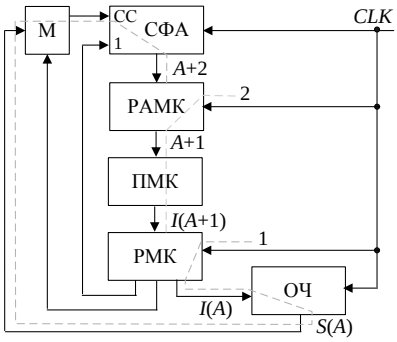
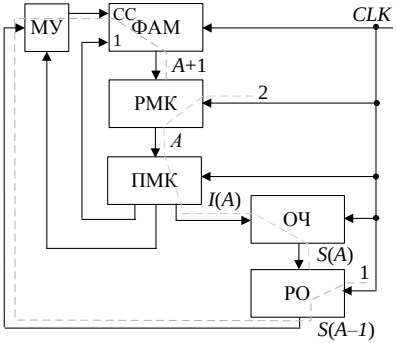


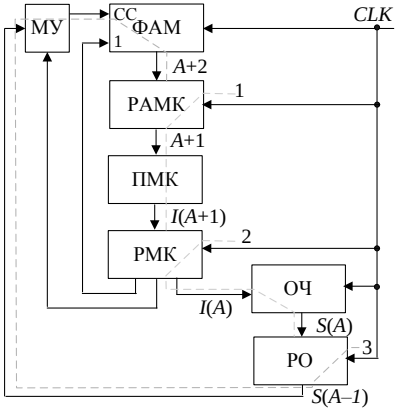
Рис. 2.38. Приклади структур процесора



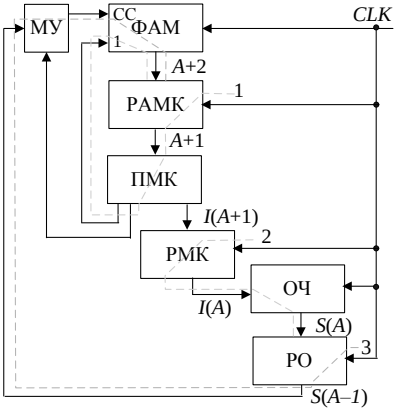
д



е



ж



з

Продовження рис. 2.38

У схемі на рис. 2.38, *б* використовується регістр мікрокоманд (РМК). В цьому випадку такт починається з запису мікрокоманди *I*(*A*) в РМК, а завершується зчитуванням із ПМК наступної мікрокоманди *I*(*A*+1).

Зменшити довжину РМК дозволяє схема, що наведена на рис. 2.38, *в*. У регістрі ознак (РО) запам'ятовуються ознаки, отримані в ОЧ. Число ознак менше за число управляючих входів ОЧ. Регістр РМК містить тільки розряди управління МУ і ФАМ.

У процесорах на рис. 2.38, *б* і 2.38, *в*, як і в процесорі на рис. 2.38, *а*, не можна поєднувати в часі виконання однієї мікрокоманди з вибіркою іншої.

Комбінуючи три базові способи організації структур, можна добитися зменшення тривалості такту за рахунок поєднання в часі деяких етапів виконання мікрокоманд. Такий спосіб виконання мікрокоманд називається конвеєрним. Приклади конвеєрної реалізації показані на рис. 2.38, *г* – рис.2.38, *з*.

Процесори на рис. 2.38, *г* – рис.2.38, *е* містять по два регістри.

Циклічний шлях проходження сигналу в схемах розділений на два незалежні шляхи. Наприклад, на 2.38, *г* один шлях містить РМК і ОЧ, а інший – РО, МУ, ФАМ і ПМК. При цьому виконання поточної мікрокоманди в ОЧ поєднується з вибіркою з ПМК наступної мікрокоманди.

Тривалість такту в цьому випадку менше, ніж при послідовному виконанні мікрокоманд. Схеми на рис. 2.38, *е* – рис.2.38, *ж* містять по три регістри і, відповідно, по три незалежні шляхи проходження сигналів.

В одному такті на різних стадіях обробки знаходяться три мікрокоманди. Тривалість такту при цьому зменшується.

Зауважимо, що при реалізації умовних переходів, коли вибірка адреси наступної мікрокоманди залежить від результату виконання попередньої, поєднання в часі виконання етапів різних мікрокоманд порушується. Умовний перехід може бути виконаний тільки в тому такті, коли необхідна інформація знаходитиметься в РО. Це призводить до збільшення числа тактів при виконанні мікропрограм.

Таким чином, розглянуті структури відрізняються тривалістю такту, кількістю тактів при реалізації заданого алгоритму і апаратурними витратами.

Зменшення тривалості такту за рахунок конвеєрної обробки мікрокоманд супроводжується зростанням апаратурних витрат і, як правило, збільшенням кількості тактів виконання мікропрограми. І з цього випливає, що вибір структури процесора повинен здійснюватися з врахуванням алгоритмів, що реалізовуються.

За однакового способу обробки мікрокоманд процесори можуть мати схемні відмінності, обумовлені системою команд.

Розглянемо структуру процесора, що реалізовуватиме певну скорочену систему команд.

Модель програміста процесора для реалізації заданої системи команд показана у табл. 2.31.

Таблиця 2.31. Модель програміста

Номер регістру	**Призначення регістрів**	
*R*0		**Регістри загального призначення (РЗП)** Застосовуються для адресації операндів у двоадресних командах. Доступ на рівні програм.
*R*1		
*R*2		
*R*3		
*R*4		
*R*5		
*R*6	Покажчик стека (ПС)	
*R*7	Лічильник команд (ЛК)	
*R*8	Регістр команд (РК)	**Робочі регістри** Застосовуються для виконання обчислень на мікропрограмному рівні, зберігання слова команди на протязі часу її виконання
*R*9	Регістр маски (РМ)	
*R*10		
*R*11		
*R*12		
*R*13	Покажчик адреси (ПА)	
*R*14		
*R*15	Регістр акумулятор (РА)	

До складу процесора входять шістнадцять регістрів $R15-R0$, які створюють надоперативний запам'ятовуючий пристрій (НОЗП). Регістри $R7-R0$ належать до регістрів загального призначення, за їх допомогою реалізуються різні способи адресації операндів, РЗП можуть використовуватися як приймачі так і джерела операндів. Доступ до цих регістрів може бути здійснений на рівні програми. Регістр $R6$ використовується як покажчик стека, $R7$ виконує функції лічильника команд (ЛК).

Регістри $R15-R8$ є робочими регістрами (*Rr*), які застосовуються для зберігання слова команди та операндів на

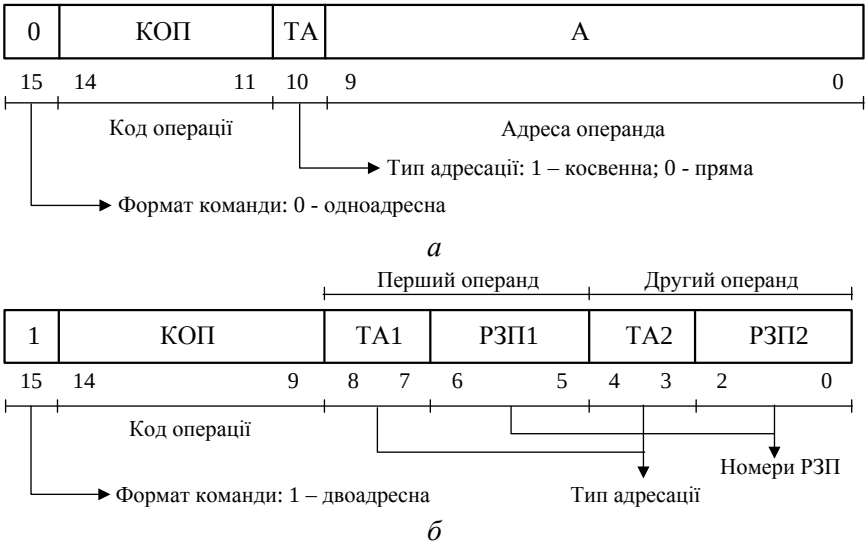
протязі всіх етапів її виконання та проміжних даних та результатів. Доступ до робочих регістрів може бути здійснений тільки на мікропрограмному рівні.

У функціональному відношенні всі команди можна класифікувати наступним чином:

– основні команди,
– команди передачі управління,
– системні команди.

До основних належать команди пересилки, зсуву, арифметичних та логічних перетворювань інформації. Команди передачі управління забезпечують розгалуження програми. Це безумовний і умовні переходи, команди умовного та безумовного переходу до підпрограмам і повернення з них. Системні команди задають режими роботи процесора, управляють перериваннями програми, тощо.

Система команд включає безадресні, одноадресні і двоадресні команди. Формати основних одноадресних і двоадресних команд показані на рис. 2.39, *а*, *б*. Адреси РЗП наведені в табл. 2.32, а приклад кодування способів адресації операндів для команд запропонованого формату – в табл. 2.33.



а

б

Рис. 2.39. Формат команди: а – одноадресної; б – двоадресної.

Таблиця 2.32. Адреси РЗП у НОЗП

РЗП	**Адреса в НОЗП**		
*R*0	0	0	0
*R*1	0	0	1
*R*2	0	1	0
*R*3	0	1	1
*R*4	1	0	0
*R*5	1	0	1
*R*6	1	1	0
*R*7	1	1	1

Таблиця 2.33. Кодування типів адресації

Код типу адресації		**Назва типу регістрової адресації**
0	0	Пряма
0	1	Косвенна
1	0	Автоінкрементна
1	1	Автодекрементна

Взагалі можуть бути реалізовані наступні типи адресації операндів:

Без застосування РЗП (одноадресні команди основної групи, команди передачі управління, команди вводу-виводу):

– пряма,
– косвенна.

Із застосуванням РЗП (команди основної групи):

– пряма,
– косвенна,
– автоінкрементна,
– автодекрементна,
– косвенна автоінкрементна,
– косвенна автодекрементна,
– індексна,
– косвенна індексна.

Схеми формування виконавчих адрес операндів при різних типах регістрової адресації показані на рис. 2.40.

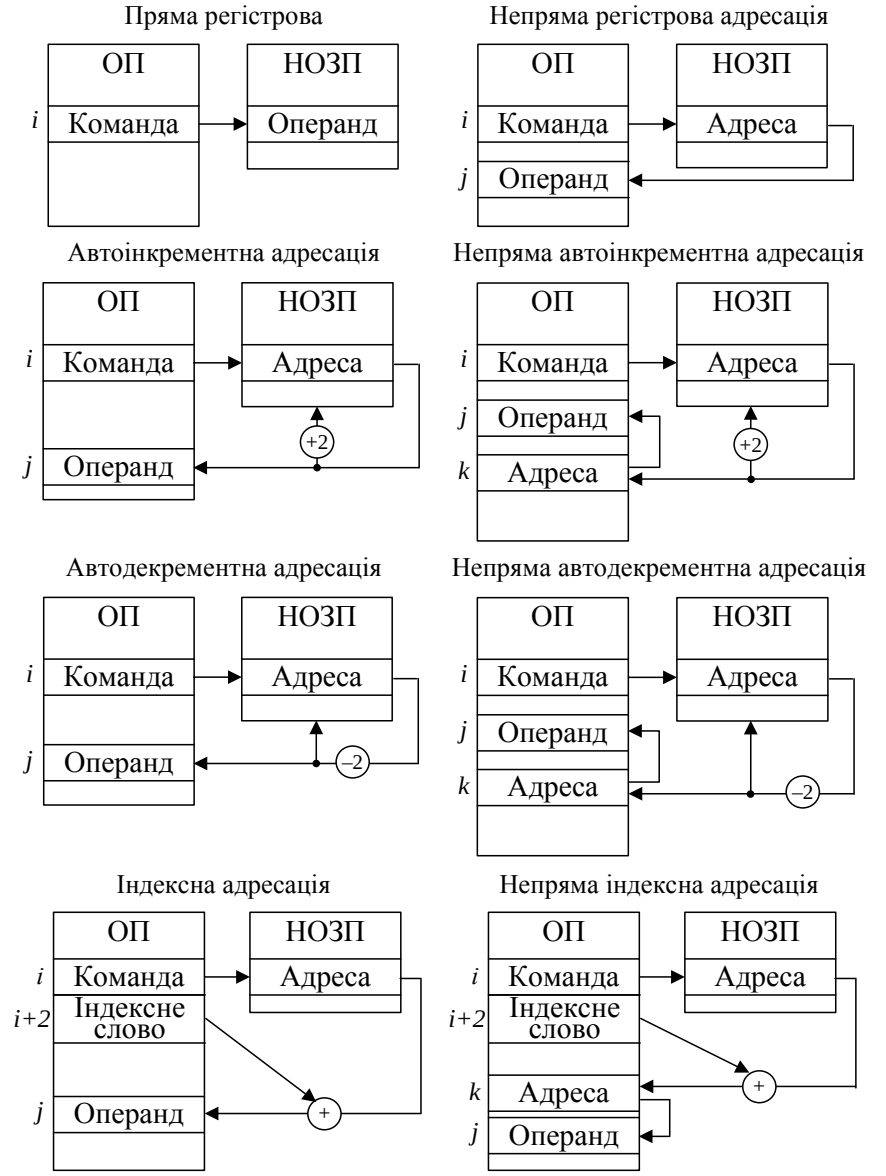


Рис. 2.40. Схеми формування виконавчих адрес операндів

Співвідношення між значеннями *i*, *k* і *j* адрес слів і загальної пам'яті можуть бути різними. При використанні способів

автоінкрементної та автодекрементної адресації (прямої і непрямої) вміст регістру РЗП, що виконує функції покажчика адреси в НОЗП, змінюється на два, якщо операндом є слово, або на одиницю, якщо операндом є байт.

Розглянемо приклади формування виконавчих адрес операндів за застосування різних способів адресації із використанням РЗП. Припустимо, що адреса регістра $R5$ – 101, в якому знаходиться операнд, вказана в молодшій тріаді коду команди (рис. 2.39). У розглянутих прикладах виконується команда інкременту.

Пряма регістрова адресація. В регістрі $R5$ знаходиться безпосередньо операнд. До вмісту $R5$ додається 1.

Непряма регістрова адресація. Вміст комірки загальної пам'яті з адресою 0010*h*, яка записана в регістрі $R5$, збільшується на одиницю.

Автоінкрементна адресація. Вміст комірки загальної пам'яті з адресою 0022*h*, яка зберігається в регістрі $R5$, збільшується на одиницю. Після чого адреса в регістрі $R5$ збільшується на два.

Непряма автоінкрементна адресація. Вміст комірки пам'яті, адреса якої 0100*h* записана в $R5$, є адресою операнду (0132*h*). Операнд розташований в пам’яті за адресою 0132*h* збільшується на одиницю.

Автодекрементна адресація. Вміст $R5$ зменшується на два. Нове значення $R5$ (j = 0770*h)* використовується як адреса операнда. Операнд розташований у пам’яті за адресою 0770*h* збільшується на одиницю.

Непряма автодекрементна адресація. Вміст $R5$ зменшується на два і використовується як адреса комірки, в якій знаходиться адреса операнда (j = 1112*h*). Операнд, записаний в загальній пам'яті за адресою 1112*h*, збільшується на одиницю.

Індексна адресація. Адреса операнда визначається додаванням вмісту $R5$ та індексного слова, яке записане в загальній пам'яті безпосередньо за словом команди (команда зберігається за адресою i = 0100*h*, а індексне слово – за адресою (i+2) = 0102*h*). Операнд у комірці пам’яті 1203*h* збільшується на одиницю.

Непряма індексна адресація. Додаванням вмісту регістру $R5$ та індексного слова формується адреса комірки, в якій зберігається адреса операнда. Операнд за адресою 0400*h* збільшується на одиницю.

Для адресації у команді може бути використаний будь-який РЗП, в тому числі і лічильник команд (регістр $R7$). Адресація з використанням лічильника команд дає можливість побудувати програму, яка не втрачає працездатність при переміщенні її в будь-яку область пам'яті. Способи адресації, що відповідають автоінкрементій, непрямій автоінкрементній, індексній та непрямій індексній, з використанням лічильника команд отримали назви безпосередня, абсолютна, відносна і непряма відносна відповідно.

У двоадресних командах типи адресації і поле адреси джерела операнда і приймача операнда можуть бути різними.

Виконання основних команд основної групи включає етапи:

1. Вибірка команди;
2. Розпакування команди
   - визначення формату команди та типу адресації;
   - обчислення виконавчих адрес і вибірка операндів;
3. Виконання операції і розміщення результату;
4. Формування адреси наступної команди.

Команда вибирається із загальної пам'яті за адресою, що знаходиться в ЛК і завантажується в регістр команд. Відповідно до типу адресації формуються виконавчі адреси операндів, здійснюється задане кодом операції перетворення інформації і засилання результату за адресою приймача. Адреса наступної команди формується в лічильнику команд збільшенням на два його вмісту.

Команди передачі управління та вводу-виводу належать до групи одноадресних команд. На етапі виконання команди умовного переходу перевіряється умова. Якщо умова виконується, то в ЛК формується нова адреса переходу. Якщо ж умова не виконується, то природний порядок виконання програми не порушується, тобто вміст ЛК збільшується на два і виконується наступна команда.

Розглянемо структуру МПС із процесором, що складається з блоку мікропрограмного управління (розділ 2.5), постійного запам’ятовуючого пристрою, шістнадцятирозрядного блоку обробки даних, побудованого на чотирьох чотирирозрядних процесорних елементах (розділ 2.2, розділ 2.4) та схеми управління станами та зсувами (розділ 2.3). Загальна структура МПС зображена на рис. 2.41.

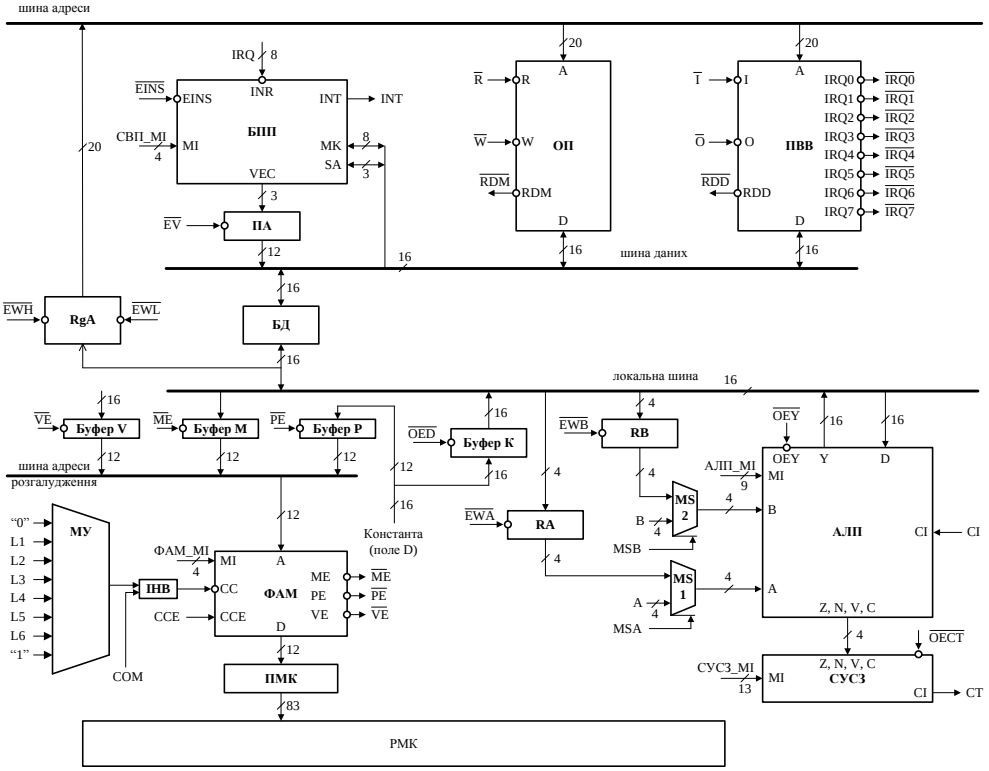


Рис. 2.41. Загальна структура мікроЕОМ

Обмін даними з основною пам'яттю (ОП) і зовнішніми пристроями (ЗП) здійснюється у програмному режимі по двоспрямованій шині даних (ШД). Операції обміну інформацією з ОП та ЗП супроводжуються сигналами запису/читання *W* (*Wright*) /*R* (*Right*) та вводу/виводу *I* (*Input*) /*O* (*Output*) відповідно. Всі управляючи сигнали кодуються у відповідних полях мікрокоманди, що формується в БМУ. Перед записом на шині адреси ША виставляється адреса комірки пам'яті або порту ЗП. На ШД виставляються дані, призначені для виводу. В якості регістра адреси та регістра даних застосовуються зовнішній регістр *RgA* та буфер даних БД. Інформація, що вводиться з ОП або ЗП приймається у процесор по ШД: спочатку на ША виставляється адреса джерела даних, після чого за відповідним управляючим сигналом дані виставляються на ШД та фіксуються в процесорі. ОП та ЗП сигналізують процесору про завершення операції фіксації даних у пристрої під час виводу даних, або готовності даних на ШД під час вводу даних за допомогою сигналів готовності *RDM* (від ОП) та *RDD* (від ЗП), що поступає для аналізу на мультиплексор умов МУ.

У склад процесору входить НОЗП, що складається з шістнадцяти шістнадцятирозрядних регістрів. Чергова команда, що вибирається з ОП на першому етапі виконання розміщується в один з робочих регістрів, відповідно моделі програміста (табл. 2.31) це регістр *R*8. За допомогою маски, що зберігається у регістрі *R*9 можна відокремити деякі розряди слова команди та виконати їх аналіз у БМУ. Так виконується аналіз розрядів ФК, ТА та КОП. Під час аналізу поля КОП розряди цього поля видаються на локальну шину (ЛШ), відповідні розряди якої приєднані до входів Буферу *M* (Додаток А, рис. А.5). Під дією управляючих сигналів КОП записується в Буфер *M*, де формується адреса переходу на мікропідпрограму виконання команди у ПМК.

Для адресації регістрів загального призначення через адресні поля команди в процесі формування виконавчих адрес застосовуються зовнішні регістри *PA* та *PB*. Інформація з адресних полів слова команди по ЛШ передається у зазначені регістри. Мультиплексори *MA* та *MB* дозволяють адресувати регістри НОЗП, як через поля команди так і через поля мікрокоманди у ПМК.

З ціллю зсувів слів, обробки ознак та формування вхідного переносу застосовується СУСЗ.

Буфер *V* БМУ застосовується для реалізації переривань. Під час виникнення переривання на входи Буферу *V* подаються вектори переривань, що є адресами мікропідпрограм обробки переривань.

Шина даних процесора має довжину 16 розрядів. Процесор може здійснювати обмін шістнадцятирозрядними словами.

Шина адреси має довжину 20 розрядів. Ємність ОП становить 1М 16-розрядних слів. Мінімальна інформаційна одиниця доступу – 16-розрядне слово, доступ до окремого байта слова неможливий. Слова в ОП адресуються тільки парними адресами.

Обмін інформацією із ЗП здійснюється за допомогою спеціальних команд вводу-виводу *IN* (*Input*) *OUT* (*Output*). Адреси регістрів ЗП (портів вводу-виводу: РС – регістру стану та РД – регістру даних) утворюють з комірками ОП єдиний адресний простір.

Під час розробки мікропрограм на першому етапі складається функціональний мікроалгоритм (Ф-мікроалгоритм) перетворення інформації, що включає узагальнені мікрооперації. Слід зазначити, що такі мікрооперації можуть бути відсутні у системі мікрооперацій застосованого ПЕ. Узагальнені мікрооперації записують змістовними термінами або операторами присвоєння. На підставі Ф-мікроалгоритму розробляється функціонально-структурний мікроалгоритм (ФС-мікроалгоритм), що враховує реальну систему мікрооперації та зв’язки, власні обраній структурі процесора. В операційних вершинах ФС-мікроалгоритму розміщуються тільки ті мікрооперації, що виконуються за один такт роботи процесора. ФС-мікроалгоритм є вихідною інформацією для розробки мікропрограм та розміщення їх у ПМК.

Розглянемо приклад розробки мікроалгоритмів для основних етапів виконання команди. Припустимо, що процесор реалізує скорочену систему команд, наведену у табл. 2.34.

Ф-мікроалгоритм основних етапів виконання команди наведений на рис. 2.42. На етапі виборки команда зчитується з ОП за адресою, що надійшла із лічильника команд у регістр РА. Далі процесор генерує сигнал читання *R* і переходить у стан очікування сигналу готовності. Вибране слово команди поступає на ШД, ОП сигналізує процесору про готовність даних сигналом *RDM* і команда приймається у РК НОЗП.

Таблиця 2.34. Система команд процесора

Двоадресні команди													
Мнемоніка	КОП	Двійковий код команди у ОП								Адреса МПП у ПМК			
		ФК	КОП				ТА1	А1	ТА2	А2	КОП		
ADD	00010	1	0 0 0 0 1				* *	* * *	* *	* * *	00	00010	00000
SUB	00100	1	0 0 1 0 1				* *	* * *	* *	* * *	00	00100	00000
MUL	00110	1	0 0 1 1 0				* *	* * *	* *	* * *	00	00110	00000
DIV	01110	1	0 0 1 1 1				* *	* * *	* *	* * *	00	01110	00000
Одноадресні команди													
Мнемоніка	КОП	Двійковий код команди у ОП								Адреса МПП у ПМК			
		ФК	КОП				ТА	А			КОП		
MOV	0011	0	0 0 1 1				*	* * * * * * * * * *			00	0011	000000
JMP	1001	0	1 0 0 1				*	* * * * * * * * * *			00	1001	000000
CJS	1000	0	1 0 0 0				*	* * * * * * * * * *			00	1000	000000
IN	1100	0	1 1 0 0				*	* * * * * * * * * *			00	1100	000000
OUT	1101	0	1 1 0 1				*	* * * * * * * * * *			00	1101	000000

На етапі розпакування команди визначається її формат та тип адресації операндів. Залежно від формату команди виконується розпакування одноадресної, або двоадресної команди. Залежно від типу адресації обчислюється виконавча адреса операндів і вибір за цією адресою операндів у робочі регістри НОЗП. За двоадресній адресації один з операндів, що адресується полем команди РЗП1 пересилається в робочий регістр НОЗП $Rr1$, а другий, що адресується через адресне поле РЗП2, – в робочий регістр $Rr2$. Для одноадресних команд в робочий регістр пересилається тільки один операнд, що адресується адресним полем А (рис. 2.42).

Приклад обчислення виконавчої адреси A і вибірки операнда у робочий регістр НОЗП для непрямої адресації показаний на рис. 2.43, де через РЗПi позначений номер регістру, що вміщує адресну інформацію.

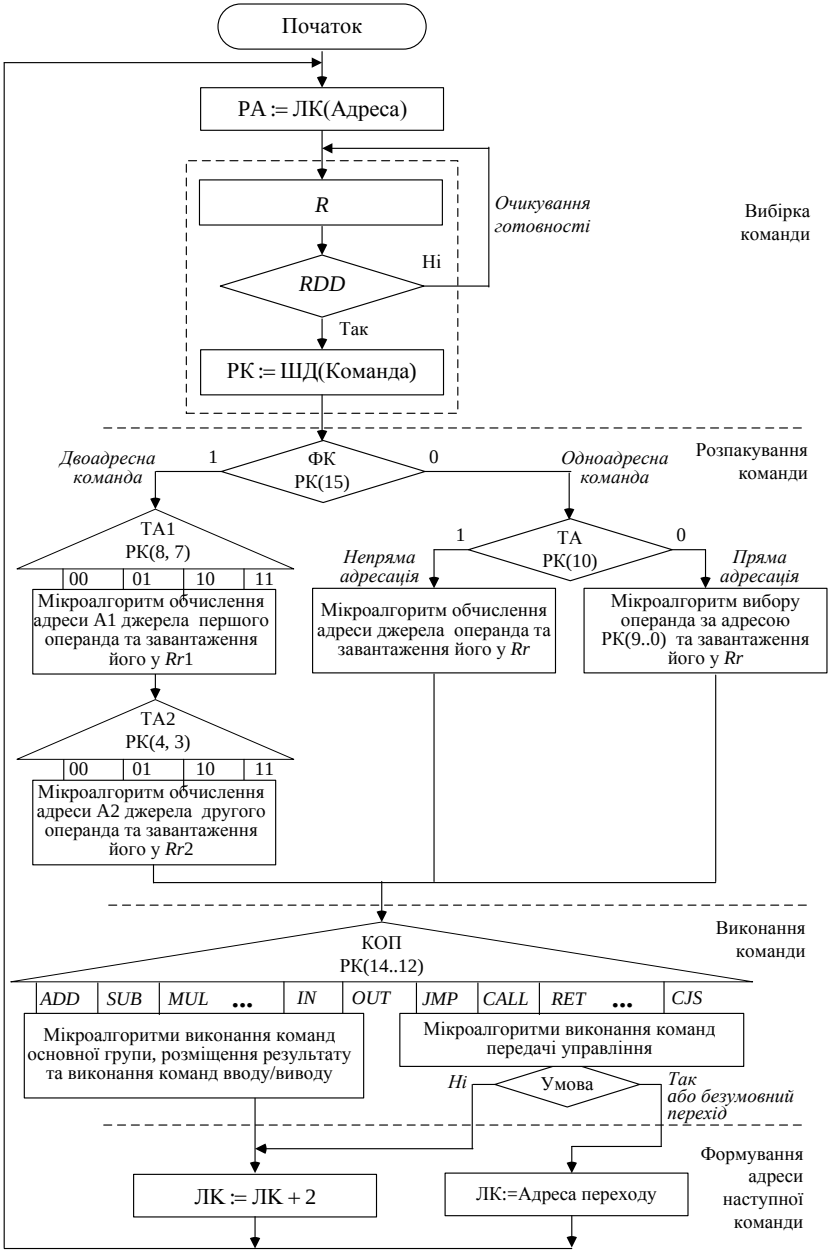


Рис. 2.42. Ф-мікроалгоритм виконання команди

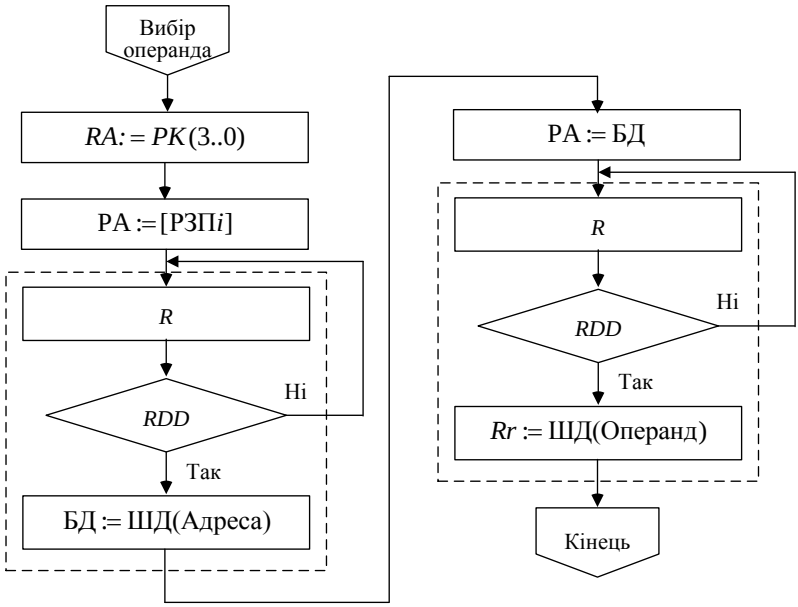


Рис. 2.43. Ф-мікроалгоритм вибірки операнда

На етапі виконання команди здійснюється перехід на мікроалгоритм виконання заданої операції відповідно коду операції. При виконанні команд передачі управління здійснюється формування адреси переходу в ЛК. При виконанні одно- і двоадресних команд основної групи здійснюється задане перетворення інформації. Результат формується в одному з робочих регістрів *Rr* , з якого можна видати результат на ШД для запису в пам'ять.

Під час прямої регістрової адресації результат записується НОЗП і застосовується у подальших обчисленнях, або за допомогою команди пересилки *MOV,* засилається в ОП. При інших способах адресації результат може бути записаний в ОП за адресою другого операнда, яка на етапі розпакування команди була збережена в РА. Слід зазначити, що мікроалгоритми виконання команд складаються проектувальниками в залежності від цільової функції МПС.

Формування адреси наступної команди за природного порядку виконання команд зводиться до збільшення на два вмісту ЛК.

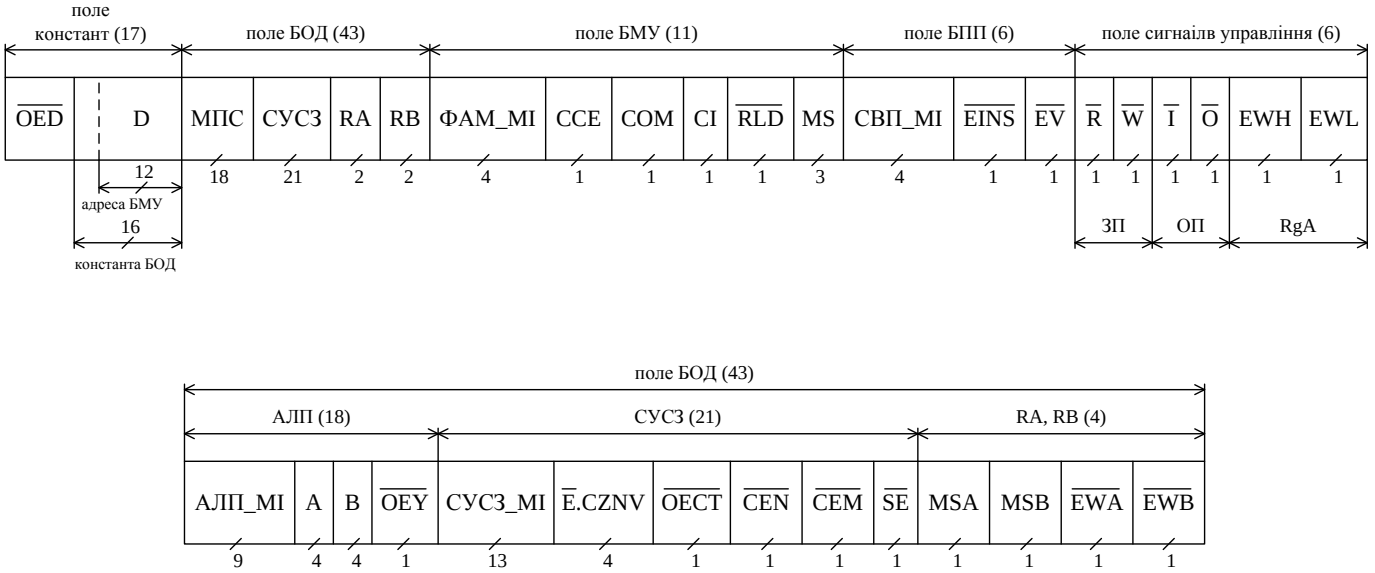


Рис. 2.44. Мікрокоманда для мікроЕОМ: а – структура мікрокоманди; б – поля мікрокоманди для управління БОД

Мікрокоманда для ЕОМ на рис. 2.44 має наступну структуру:

Поле констант:

OED – дозвіл видачі константи;
D – дані - константа, що видається на ШД;

Поле управління БОД:

АЛП_МІ – мікроінструкція;
A – адреса регістра НОЗП за каналом *A*:;
B – адреса регістра НОЗП за каналом *B;*
OEY – дозвіл видачі результату на ШД:

Поле управління СУСЗ:

СУСЗ_МІ – мікроінструкція;
EZ – дозвіл запису ознаки *Z*;
EN – дозвіл запису ознаки *N*;
EC – дозвіл запису ознаки *C*;
EV – дозвіл запису ознаки *V*;
OECT – дозвіл видачі умови;
CEM – дозвіл запису всіх ознак в *RM*;
CEN – дозвіл запису всіх ознак в *RM*;
SE – дозвіл зсуву;

Поле управління БМУ:

ФАМ_МІ – мікроінструкція;
CCE – дозвіл аналізу умов;
COM – інвертування умови;
CI – вхідне перенесення для схеми інкременту;
RLD – дозвіл запису в регістр адреси/лічильника;
MS – управління мультиплексором умов;

Поле управління регістрами і мультиплексорами:

MSA – управління мультиплексором *MA*;
MSB – управління мультиплексором *MB*;
EWA – дозвіл запису в *PAA*;
EWB – дозвіл запису в *PAB*;
EWH – дозвіл запису в старші розряди *PA*;
EWL – дозвіл запису в молодші розряди *PA*;

Поле управління ОП і ВУ:

R – сигнал читання даних з ОП;
W – сигнал запису даних в ОП;

I – сигнал вводу даних з ЗП
O – сигнал виводу даних на ЗП

При розробці ФС-мікроалгоритмів узагальнені мікрооперації у Ф-мікроалгоритмі замінюють на мікрооперації, що входять до складу системи мікрооперацій ЕОМ.

Розглянемо фрагменти ФС-мікроалгоритмів, що забезпечують наступні етапи виконання команди – вибірка команди з пам'яті (рис. 2.45), розпакування команди (рис. 2.46) та перехід на мікропідпрограму виконання операції (рис. 2.47), а мікропрограма в кодах мікрокоманд, що відповідає наданим фрагментам ФС-мікроалгоритмів приведена на рис. 2.49.

Мікропрограма розміщується в ПМК розпочинаючи з нульової адреси. Адреси мікрокоманд представлені в шістнадцятирозрядній системі числення. Активний рівень сигналу готовності пам'яті (*RDM*) є нульовим. Функціональне застосування регістрів НОЗП відповідає моделі програміста поданій у табл. 2.31.

Фрагмент ФС-мікроалгоритму, що забезпечує розпаковку відповідних форматів команд, представлений на рис. 2.46. Розряд, що визначає за формат команди у слові команди співпадає зі знаковим розрядом, тому визначити формат команди можна аналізуючи знак слова команди. Під час визначення типу адресації також здійснюється аналіз відповідних розрядів слова команди (табл. 2.34). Аналізується розряди, виділені за допомогою масок (маски завантажуються в регістр *R*9). Маски зазвичай добираються наступним чином: в розряд маски відповідний розряду слова команди, що аналізується, записують одиницю, інші розряди дорівнюють нулю. Між регістром маски та регістром команди виконують мікрооперацію логічного множення (кон'юнкція). При цьому результат виконання мікрооперації в АЛБ нікуди не записується – приймач *NIL*). В ІС ВС1 під час проходження інформації через АЛБ формується ознака рівності результату нулю *ZO*. За рівності нулю результату виконання мікрооперації слід зробити вивід, що аналізований розряд слова команди дорівнює нулю, та навпаки. На рис. 2.46 зображені гілки мікроалгоритму відповідні розпакуванню одноадресної команди і прямої адресації операндів.

Галуження на декілька напрямів при розробці ФС-мікроалгоритмів зручно здійснювати з використанням

мікроінструкції переходу *JMAP* блоку мікропрограмного управління. В цьому випадку перехід здійснюється за один такт, а адреса переходу визначається інформацією у Буфері *M*. Для аналізу поля КОП слова команди до входів Буферу *M* можна підключити відповідні розряди ЛШ, що забезпечить аналіз коду операції для будь-якої команди. За такого способу галуження адреси переходів на мікропідпрограми виконання операцій матимуть вигляд, показаний в табл. 2.34.

Фрагмент ФС-мікроалгоритму переходу на виконання одноадресної команди за кодом операції показаний на рис. 2.47. Перед виконанням функції *JMAP* в над вмістом РК виконується мікрооперація, наприклад, додавання нуля, результат нікуди не зберігається, але за встановлення сигналу $\overline{OEY}$ видається на ШД, звідки поступає у Буфер *M*.

Мікропрограми, розроблені із застосуванням символічного мвкроасемблеру та у кодах мікрокоманд, що реалізують мікроалгоритми на рис. 2.45 – рис. 2.47 приведені на рис. 2.48, рис. 2.49 відповідно.

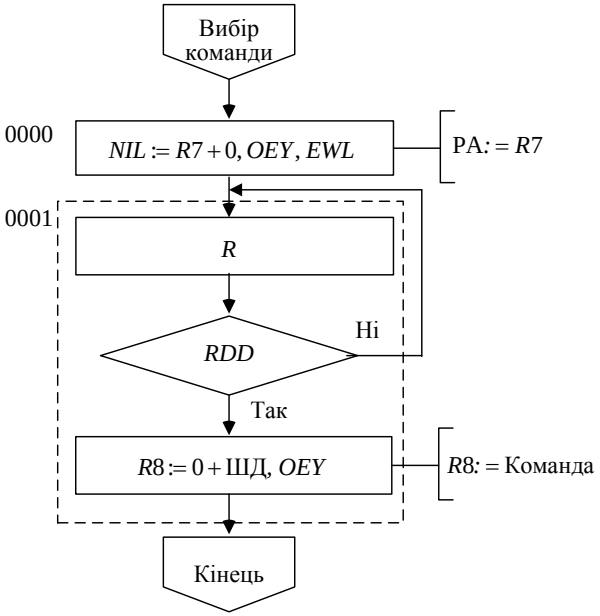


Рис. 2.45. ФС-мікроалгоритм вибірки команди з ОП

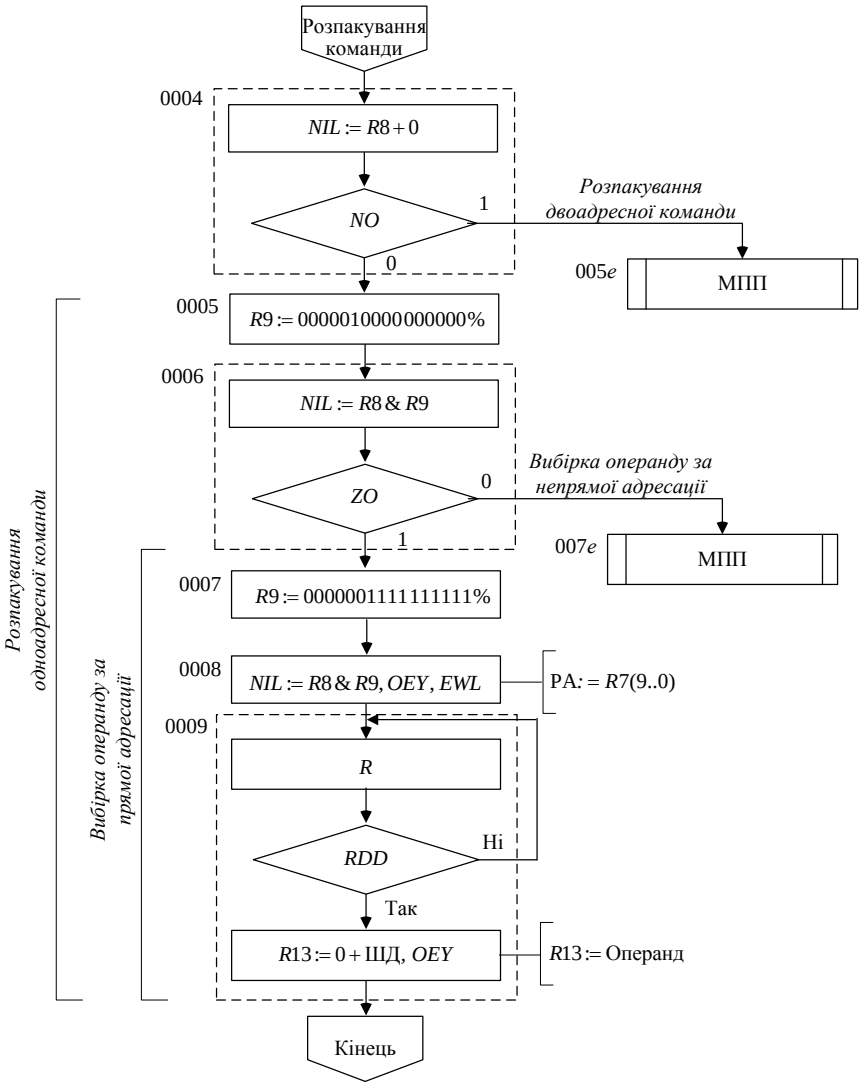
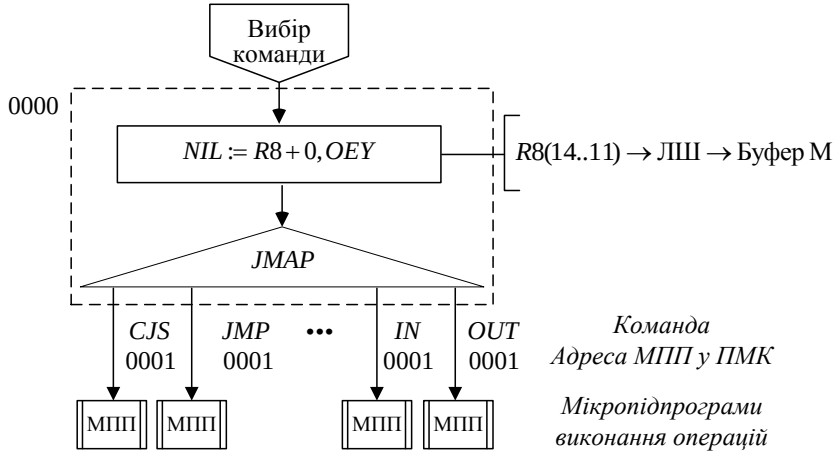


Рис. 2.46. ФС-мікроалгоритм розпакування одноадресної команди

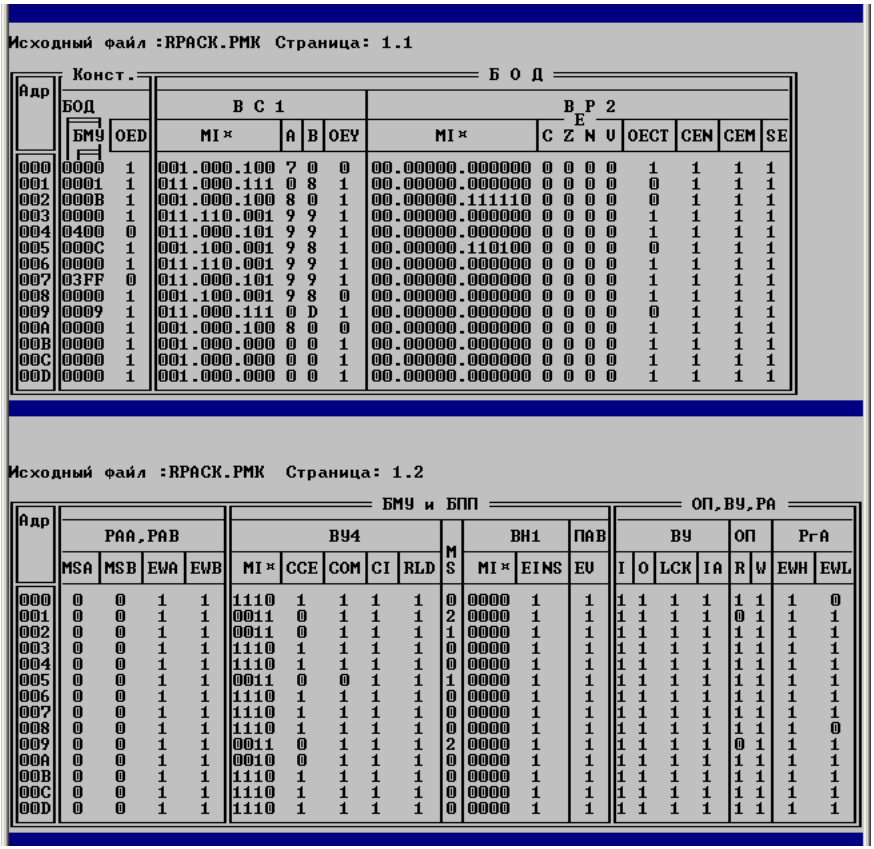


Р

ис. 2.47. ФС-мікроалгоритм переходу на МПП за кодом операції

```
      /-- Область налагодження зв'язків
      link l1: ct
      link l2:rdm
      link ewh:16
      link m:z,z,z,15,14,13,12,11,z,z,z,z
      /-- Видача адреси команди у РА
0000  {add nil,r7,z;oey;ewl;}
      /-- авантаження команди у РК
0001  {r;cjp rdm,cp;add r8,bus_d,z;}
      /-- Визначення ФК
0002  {add nil,r8,z;cjp no,dfk;
      /-- Визначення ТА}
0003  {xor r9,r9,r9;}
0004  {add r9,r9,400h;}
0005  {and nil,r9,r8;cjp not zo,kad;}
      /-- Завантаження операнда
0006  {xor r9,r9,r9;}
0007  {add r9,r9,3ffh;}
0008  {and nil,r9,r8;oey;ewl;}
0009  {r;cjp rdm,cp;add r13,bus_d,z;}
      /-- Перехід по КОП
000A  {add nil,r8,z;oey;jmap;}
000B  {dfk}/--МПП розпакування двоадресної команди
000C  {kad}/--МПП завантаження для непрямої адресації
000D  {}
```

Рис. 2.48. Мікропрограма виконання одноадресної команди



Исходный файл :RPACK.PMK Страница: 1.1

Адр	Конст. БОД БМУ	Конст. БОД OED	БОД ВС1 MI*	БОД ВС1 A	БОД ВС1 B	БОД ВС1 OEY	БОД ВР2 MI*	БОД ВР2 C	БОД ВР2 Z	БОД ВР2 N	БОД ВР2 V	БОД ВР2 OECT	БОД ВР2 CEN	БОД ВР2 CEM	БОД ВР2 SE
000	0000	1	001.000.100	7	0	0	00.00000.000000	0	0	0	0	1	1	1	1
001	0001	1	011.000.111	0	8	1	00.00000.000000	0	0	0	0	0	1	1	1
002	000B	1	001.000.100	8	0	1	00.00000.111110	0	0	0	0	0	1	1	1
003	0000	1	011.110.001	9	9	1	00.00000.000000	0	0	0	0	1	1	1	1
004	0400	0	011.000.101	9	9	1	00.00000.000000	0	0	0	0	1	1	1	1
005	000C	1	001.100.001	9	8	1	00.00000.110100	0	0	0	0	0	1	1	1
006	0000	1	011.110.001	9	9	1	00.00000.000000	0	0	0	0	1	1	1	1
007	03FF	0	011.000.101	9	9	1	00.00000.000000	0	0	0	0	1	1	1	1
008	0000	1	001.100.001	9	8	0	00.00000.000000	0	0	0	0	1	1	1	1
009	0009	1	011.000.111	0	D	1	00.00000.000000	0	0	0	0	0	1	1	1
00A	0000	1	001.000.100	8	0	0	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	0000	1	001.000.000	0	0	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

Исходный файл :RPACK.PMK Страница: 1.2

Адр	БМУ и БПП PAA,PAB MSA	PAA,PAB MSB	PAA,PAB EWA	PAA,PAB EWB	БУ4 MI*	БУ4 CCE	БУ4 COM	БУ4 CI	БУ4 RLD	MS	ВН1 MI*	ВН1 EINS	ПАВ EV	ОП,ВУ,РА ВУ I	ВУ O	ВУ LCK	ВУ IA	ОП R	ОП W	РгА EWH	РгА EWL
000	0	0	1	1	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	0
001	0	0	1	1	0011	0	1	1	1	2	0000	1	1	1	1	1	1	0	1	1	1
002	0	0	1	1	0011	0	1	1	1	1	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	0011	0	0	1	1	1	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	1110	1	1	1	1	0	0000	1	1	1	1	1	1	1	1	1	0
009	0	0	1	1	0011	0	1	1	1	2	0000	1	1	1	1	1	1	0	1	1	1
00A	0	0	1	1	0010	0	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	1110	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

Рис. 2.49. Мікропрограма в кодах мікрокоманд виконання одноадресної команди