

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний авіаційний університет

В.І. ЖАБІН, І.А. ЖУКОВ, В.В. ТКАЧЕНКО, І.А.КЛИМЕНКО

МІКРОПРОЦЕСОРНІ СИСТЕМИ

Навчальний посібник

Київ 2009

УДК 004.31 (076.5)

Укладачі: В.І. ЖАБІН, І.А. ЖУКОВ, В.В. ТКАЧЕНКО,
І.А.КЛИМЕНКО

Рецензенти

Г.М. ЛУЦЬКИЙ, д-р техн. наук, проф.

(Національний технічний університет України «КПІ»)

В.Ф. ЄВДОКИМОВ, д-р техн. наук, проф.

(Інститут проблем моделювання в енергетиці ім. Г.Є.Пухова НАН України)

І.А. ДИЧКА, д-р техн. наук, проф.

(Національний технічний університет України «КПІ»)

Затверджено на засіданні Вченої ради Інституту комп'ютерних технологій Національного авіаційного університету від 17 листопада 2008 року

Мікропроцесорні системи: навчальний
Ж 752 посібник/Уклад.: В.І.Жабін, І.А.Жуков, В.В.Ткаченко
І.А.Клименко,.- К.:НАУ, 2009.- 475 с.

Містить лекційний матеріал, завдання, рекомендації і приклади до виконання практичних робіт, розрахунково-графічних робіт, курсового проектування, контрольні питання та задачі для самостійного розв'язування та типові завдання до рейтингових контролів.

Призначений для студентів напрямів «Комп'ютерна інженерія» та «Комп'ютерні науки»

ЗМІСТ

Передмова	3
Перелік скорочень	5

ТЕОРЕТИЧНА ЧАСТИНА

1. Введення в архітектуру мікропроцесорних систем

1.1. Функціональна класифікація мікропроцесорів.....	9
1.2. Різновиди архітектури мікропроцесорів	11
1.3. Організація обчислювальних процесів в мікропроцесорних системах...	15

2. Мікропроцесори на основі секціонованих інтегральних схем

2.1. Загальні відомості.....	23
2.2. Процесорний елемент.....	23
2.3. Схема управління станами та зсувами	34
2.3.1. Загальна структура та принцип функціонування	34
2.3.2. Схема управління	39
2.3.3. Блок обробки ознак	39
2.3.4. Виконання операцій з вмістом регістрів станів	40
2.3.5. Блок перевірки умови	48
2.3.6. Блок управління переносам	51
2.3.7. Блок управління зсувами	52
2.3.8. Виконання операції нормалізації	61
2.3.9. Реалізація переривань	65
2.4. Блоки обробки даних на секціонованому комплекті інтегральних схем	66
2.5. Блоки мікропрограмного управління	72
2.6. Блоки пріоритетних переривань	110
2.6.1. Структурна організація блока пріоритетних переривань	110
2.6.2. Система мікрокоманд схеми векторних переривань	110
2.7. Проектування мікропроцесорних систем на секціонованому комплекті інтегральних схем	116

3. Однокристальний мікроконтролер KP1816BE48

3.1. Загальна характеристика	139
3.2. Архітектура мікроконтролера KP1816BE48	139

3.2.1. Умовне графічне позначення мікросхеми	139
3.2.2. Структура мікроконтролера	141
3.3. Основні режими роботи мікроконтролера KP1816BE48	151
3.3.1. Ініціалізація системи	151
3.3.2. Режими роботи з пам'яттю	152
3.3.3. Підключення додаткових портів	159
3.3.4. Підключення програмованого периферійного адаптера K580BB55	161
3.3.5. Підключення програмованого зв'язкового адаптера K580BB51	169
3.4. Система команд мікроконтролера KP1816BE48	184
3.4.1. Формат команд	184
3.4.2. Система команд	184
3.5. Розробка програм обробки даних	200
3.5.1. Виконання команд передачі даних	200
3.5.2. Виконання команд арифметичних та логічних операцій	201
3.5.3. Виконання команд передачі управління	204
3.5.4. Розробка підпрограм виконання складних арифметичних операцій	205
3.5.5. Розробка програм управління	217
4. Однокристальний мікроконтролер KP1816BE51	
4.1. Загальна характеристика	225
4.2. Архітектура мікроконтролера KP1816BE51	225
4.2.1. Умовне графічне позначення мікросхеми	225
4.2.2. Структура мікроконтролера	227
4.2.3. Організація пам'яті	234
4.2.4. Таймери/лічильники	241
4.2.5. Порти вводу/виводу	245
4.2.6. Блок послідовного інтерфейсу і переривань	246
4.2.7. Система переривань	249
4.3. Система команд мікроконтролера KP1816BE51	251
4.3.1. Формат команд	251
4.3.2. Способи адресації операндів	270
4.3.3. Формування ознак результату	275
4.3.5. Виконання команд передачі даних	277
4.3.6. Виконання операцій зі стеком	280
4.3.7. Виконання арифметичних операцій	281
4.3.8. Виконання команд логічних операцій	284

4.3.9. Виконання операцій з бітами	285
4.3.10. Виконання команд передачі управління	288
4.3.11. Виконання команди непрямого переходу	290
4.3.12. Програмування послідовного порту	290
4.3.12. Арифметичні операції з плаваючою комою	293
4.3.13. Обчислення складних функцій	300
4.4. Схеми підключення мікросхем до мікроконтролера	308
5 Система переривань	
5.1. Загальні поняття	313
5.2. Умове графічне позначення мікросхем	319
5.3. Реалізація системи переривань в мікропроцесорних системах	322
ПРАКТИКУМ	
6 МОДУЛЬ 1. Мікропроцесори на основі секціонованих інтегральних схем	
6.1. Лабораторний практикум	333
6.2. Задачі для самостійного розв'язання	338
7 МОДУЛЬ 2. Однокристальний мікроконтролер КР1816ВЕ48	
7.1. Лабораторний практикум	342
7.2. Задачі для самостійного розв'язання	363
8 МОДУЛЬ 3. Однокристальний мікроконтролер КР1816ВЕ51	
8.1. Лабораторний практикум	372
8.2. Задачі для самостійного розв'язання	407
9 МОДУЛЬ 4. Курсовий проект	
Перелік літератури	423
Додаток А. Вказівки до використання мікроасемблера	425
Додаток Б. Моделюючий комплекс <i>SCM1</i> МК48	443
Додаток В. Моделюючий комплекс <i>SCM2</i> МК51	450
Додаток Г. Учбово-налагоджувальний стенд	464
Додаток Д. Приклади елементів цифрової техніки	459
Додаток Є. Зразки документів для оформлення курсового проекту	467

ПЕРЕЛІК СКОРОЧЕНЬ

АЛБ	– Арифметико-логічний блок
АЛП	– Арифметико-логічний пристрій
БОД	– Блок обробки даних
ВІС	– Велика інтегральна схема
ЕОМ	– Електронна обчислювальна машина
ЗП	– Зовнішній пристрій
ЗПД	– Зовнішня пам'ять даних
ЗПП	– Зовнішня пам'ять програм
ІС	– Інтегральна схема
КПДП	– Контролер прямого доступу до пам'яті
КПП	– Контролер пріоритетних переривань
МК	– Мікрокоманда
МК48	– Мікроконтроллер КР1816ВЕ48
МК51	– Мікроконтроллер КР1816ВЕ51
МП	– Мікропроцесор
МПС	– Мікропроцесорна система
НВІС	– Надвелика інтегральна схема
НОЗП	– Надоперативний запам'ятовуючий пристрій
ОЗП	– Оперативний запам'ятовуючий пристрій
ОП	– Операційний пристрій
ПД	– Пам'ять даних
ПЕ	– Процесорний елемент
ПЗП	– Постійний запам'ятовуючий пристрій
ПП	– Пам'ять програм
ПУ	– Пристрій управління
РЗП	– Регістри загального призначення
РПД	– Резидентна пам'ять даних
РПП	– Резидентна пам'ять програм
СУ	– Схема управління
СУСЗ	– Схема управління станами та зсувами
СШ	– Системна шина
ТТЛ	– Транзисторно-транзисторна логіка
УС	– Управляючі сигнали
ФАМ	– Формувач адреси мікрокоманди
ША	– Шина адреси
ШД	– Шина даних
ШУ	– Шина управління

3. ОДНОКРИСТАЛЬНИЙ МІКРОКОНТРОЛЕР КР1816ВЕ48

3.1. Загальна характеристика

ІС виду 1816ВЕ призначені для побудови восьми розрядних МПС, здатних виконувати фіксоване число програм, закладених в постійну пам'ять. Іншою назвою вказаних ІС являється мікроконтролер (МК), який визначає основну область їх застосування – управління різноманітними технічними об'єктами.

Серія ІС КМ1816 включає декілька сімейств мікроконтролерів. В даному розділі в якості базової розглядається ІС КМ1816ВЕ48. Логічна організація, система команд і засоби введення-виведення інформації вказаної БІС аналогічні ІС ВЕ35, ВЕ39 та ВЕ49 цієї ж серії. Всі БІС сімейства пристосовані до ефективного рішення задач управління та регулювання. Відмінність їх складається в організації резидентної пам'яті програм (РПП), об'ємі внутрішньої пам'яті даних (РПД) і частоті тактових сигналів (табл.3.1.).

Таблиця 3.1. Параметри мікросхем

МК	Об'єм РПП, байт	Тип ВПП	Об'єм РПД, байт	Частота тактування
ВЕ35	–	–	64	6 МГц
ВЕ48	1К	УФППЗУ	64	6 МГц
ВЕ39	–	–	128	11 МГц
ВЕ49	2К	ПЗУ	128	11 МГц

3.2. Архітектура мікроконтроллера КР1816ВЕ48

3.2.1. Умовне графічне позначення мікросхеми

Умовно графічне позначення мікросхеми МК48 наведено на рис. 3.1. Мікроконтролер конструктивно розміщений в корпусі з сорока виводами, сигнали на яких електрично-сумісні з елементами ТТЛ.

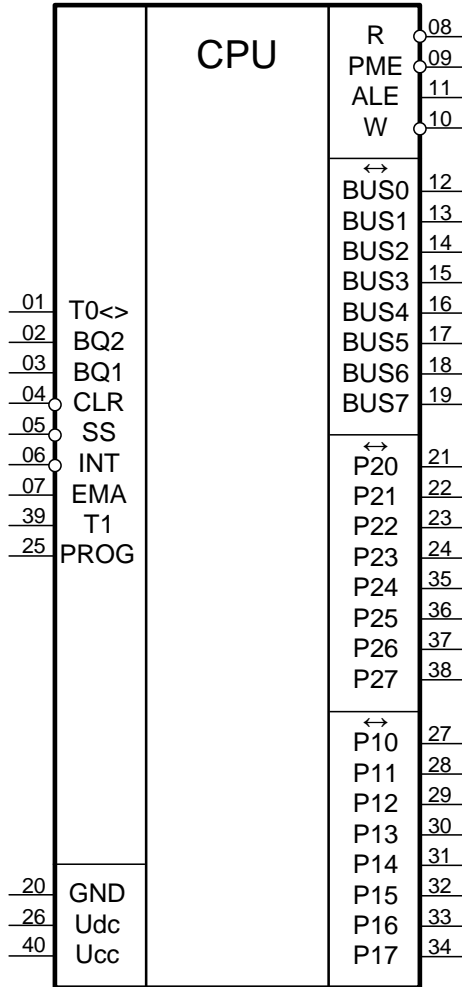


Рис. 3.1. Умовно графічне позначення мікроконтролера KP1816BE48

Нижче приводяться символічні імена виходів мікросхеми.

- T0* – вхід, що тестується командами *JT0*, *JNT0*; вихід тактового сигналу *CLK*;
- BQ1* – вхід для підключення зовнішнього джерела синхронізації (для серії KP1830) або кварцового резонатора;
- BQ2* – вхід для підключення зовнішнього джерела синхронізації (для серії KP1830) або кварцового резонатора;
- CLR* – скидання;

<i>SS</i>	– покрокове виконання програм;
<i>INT</i>	– переривання;
<i>EMA</i>	– встановлення режиму роботи з зовнішньою пам'яттю програм;
<i>T1</i>	– вхід, що тестується командами <i>JT1</i> , <i>JNT1</i> ; вхід зовнішніх подій;
<i>PROG</i>	– вихід стробу для розширювача вводу/виводу;
<i>R</i>	– читання зовнішньої пам'яті даних;
<i>PME</i>	– дозвіл читання зовнішньої пам'яті програм;
<i>ALE</i>	– строб адреси зовнішньої пам'яті;
<i>W</i>	– запис у зовнішню пам'ять даних;
<i>GND</i>	– загальний;
<i>Ucc</i>	– напруга живлення +5В;
<i>UDC</i>	– додаткова напруга живлення +5В;
<i>BUS0 – BUS7</i>	– шина даних; порт <i>BUS</i> ;
<i>P10 – P17</i>	– восьмирозрядний порт вводу-виводу;
<i>P20 – P27</i>	– восьмирозрядний порт вводу-виводу;

3.2.2. Структура мікроконтролера KP1816BE48

Структурна схема МК48 наведена на рис. 3.2. Мікроконтролер містить резидентну пам'ять програм (РПП), резидентну пам'ять даних (РПД); пристрій управління і синхронізації, до складу якого входить лічильник команд, реєстр команд і реєстри ознак; арифметико-логічний пристрій, до складу якого входить АЛБ, акумулятор і реєстри; реєстр слова стану програми; таймер/лічильник. Обмін даними здійснюється через порти *P1*, *P2* і *BUS*.

Модель програміста МК48 зображена на рис. 3.3. На моделі програміста застосовуються наступні умовні позначення:

<i>PC</i>	– лічильник команд;
<i>T</i>	допоміжний реєстр;
<i>A</i>	– акумулятор,
<i>TCNT</i>	– таймер/лічильник;
<i>RSW</i>	– реєстр слова стану програми;
<i>MB</i>	– ознака банку пам'яті;
<i>TF</i>	– ознака переповнювання таймера;
<i>F1</i>	– ознака користувача;
<i>C</i>	– ознака переносу;
<i>AC</i>	– ознака додаткового переносу;

<i>F0</i>	– ознака користувача;
<i>RB</i>	– ознака банку регістрів;
<i>SP</i>	– покажчик стеку;
<i>P1</i>	– порт вводу/виводу
<i>P2</i>	– порт вводу/виводу
<i>BUS</i>	– порт вводу/виводу

Акумулятор

Акумулятор *A* являється восьмирозрядним регістром, який крім вказівника адреси, використовується в якості приймача або джерела операнда.

Арифметико-логічний пристрій

До складу АЛП входять наступні блоки: комбінаційна схема обробки байтів, регістри *T*, регістр-акумулятор *A*, схема десяткового коректора і схема формування ознак .

Акумулятор використовується як регістр операнда і регістра результату. Регістр тимчасового зберігання операнда *T1* програмно недоступний і використовується для тимчасового зберігання другого операнда при виконанні двооперандних команд. Комбінаційна схема АЛП може виконувати наступні операції: складання байтів з перенесенням або без нього; логічні операції І, АБО та ВИКЛЮЧНЕ АБО; інкремент, декремент, інверсія, циклічний зсув вліво і вправо із встановленням (або без встановлення) ознаки переносу, обмін тетрадами в байті; десяткова корекція вмісту акумулятора.

Під час виконання команди $\mathcal{J}b$ (де $b = \overline{0,7}$) в АЛБ формується ознака Bb , яка відповідає значенню відповідного розряду акумулятора (рис. 3.3). Під час виконання команд \mathcal{JZ} , \mathcal{JNZ} , \mathcal{DJNZ} формується ознака z нульового вмісту акумулятора або зазначеного у команді регістру. За значеннями ознак Bb і z здійснюється розгалуження програм. Тригери для зберігання ознак Bb і z відсутні тобто після виконання команд ознаки не запам'ятовуються.

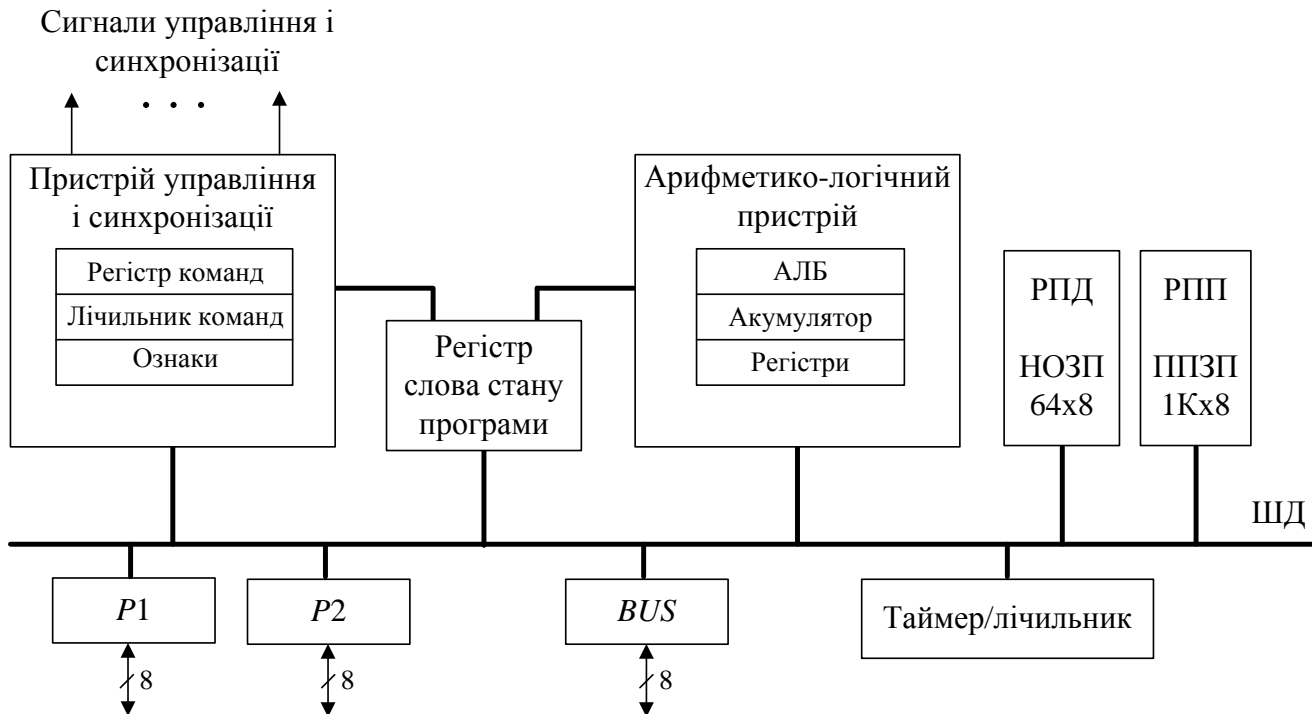


Рис. 3.2. Структурна схема мікроконтролера KP1816BE48

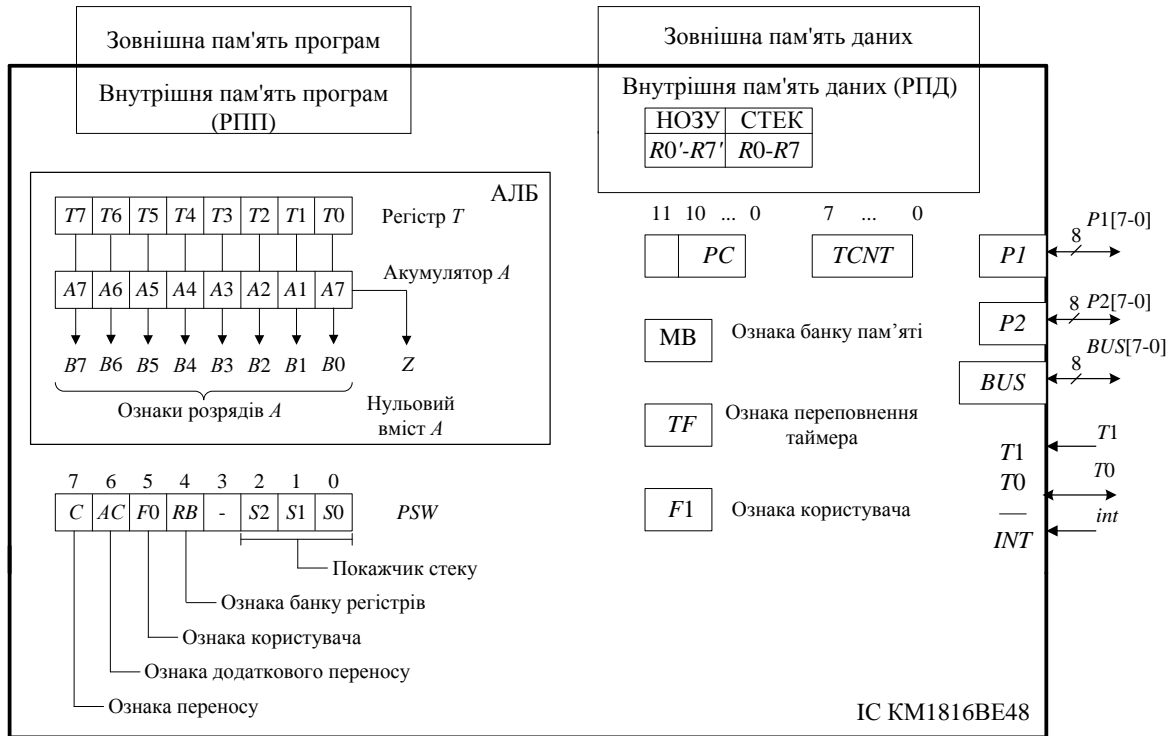


Рис. 3.3. Модель програміста

При виконанні операцій обробки даних в АЛБ виробляються ознаки, які формуються на комбінаційній схемі і не фіксуються на тригерах, за винятком ознаки переносу *C*.

Ознака додаткового переносу *AC* встановлюється під час переносу з молодшої тетради в старшу. Ознаки переносу і допоміжного переносу фіксуються на тригерах, що входять до складу регістра слова стану програми *PSW*. Формат регістра *PSW* показаний на рис. 3.3. Окрім перерахованих ознак логіка умовних переходів МК оперує прапорами *F0* і *F1*, функціональне призначення яких визначається розробником; прапором переповнювання таймера *TF*, сигналами на входах *T0* і *T1*. Програмістом можуть бути також використані ознаки робочого банку регістрів *RB* і вибраного банку зовнішньої пам'яті програм *MB*. Крім того, логікою переходів після закінчення кожного машинного циклу обпитується ще одна ознака, а саме ознака дозволу/заборони переривання.

Лічильник команд

Лічильник команд *PC* має довжину дванадцять розрядів. Після вибірки чергової команди вміст *PC* збільшується на одиницю. Перенос при цьому розповсюджується тільки від нульового до десятого розряду.

Старший розряд *PC[11]* виконує спеціальну функцію і визначається ознакою банку пам'яті програм *MB*, яка встановлюється програмно, за допомогою команд *SFL MBO* і *SEL MB1*. Програмні засоби для перевірки ознаки банку пам'яті *MB* відсутні.

Пам'ять програм

Пам'ять програм і пам'ять даних в МК48 розділені.

Пам'ять програм розділяється на резидентну, розташовану всередині ІС, та зовнішню, для реалізації якої необхідні додаткові ІС пам'яті. Максимальний адресний простір пам'яті програм складає 4 Кб (рис. 3.4). Резидентна пам'ять програм (РПП) представляє собою перепрограмоване ПЗУ(ППЗУ) об'ємом 1 Кб (адреса від 0 до 1023).

Адреси 0, 0003 та 0007 мають спеціальне призначення. З адреси 0 починається виконання програми за системним скиданням. Комірка 0003 призначена для зберігання початкової адреси підпрограми обслуговування зовнішнього переривання, а комірка 0007 – зберігання початкової адреси підпрограми обробки переривання від таймера/лічильника.

Пам'ять програм розглядається як два банки – нульовий банк програм і перший банк програм. Якщо встановлюється розряд лічильника команд $PC[11] = 0$, то вибір слів здійснюється із нульового банку пам'яті (адреси від 0 до 2047), і, якщо $PC[11] = 1$ – із першого банку пам'яті (адреси від 2047 до 4095). Вибір банку пам'яті здійснюється командами $SEL\ MBO$ та $SEL\ MB1$, які встановлюють ознаку MB вибору банку пам'яті.

Окрім розділення на банки програм, пам'ять програм поділяється на сторінки об'ємом 256 байт. Це пов'язано з тим, що команди умовних переходів модифікують тільки вісім молодших розрядів адреси, тобто забезпечують перехід всередині сторінки. При переході до підпрограм обслуговування переривань автоматично встановлюється в нуль розряд $PC[11]$ лічильника команд. У зв'язку з цим підпрограми обслуговування переривань повинні розміщуватися в нульовому банку пам'яті.

Способи адресації операндів в пам'яті програм:

- безпосередня;
- непряма з використанням акумулятора.

В першому випадку операнд розміщується в байті, наступному за кодом команди. За непрямої адресації, в якості покажчика адреси операнда застосовується акумулятор A . Акумулятор містить адресу чергової сторінки пам'яті ($MOV\ A, @A$), або третьої сторінці ($MOV\ 3\ A, @A$).

Пам'ять даних

Пам'ять даних розділяється на внутрішню і зовнішню. Внутрішня пам'ять даних представляє собою ОЗП ємністю 64 байти (рис. 3.5). Пам'ять даних містить два банки реєстрів загального призначення. Банк реєстрів 0 включає реєстри $R0-R7$ з адресами 0 – 0007, а банк реєстрів 1 – реєстри $R0-R7$, які мають адреси 0024 – 0031. Вибір банку реєстрів здійснюється командами $SEL\ RBO$ та $SEL\ RB1$, які встановлюють ознаку RB , що знаходиться в четвертому розряді PSW .

Спеціальна команда для перевірки RB відсутня, але ознаку можна проаналізувати, перенісши вміст PSW в A і виконати перехід за ознакою $B4$, яка перевіряється командою $J\ B4$. Комірки з адресами 0008 – 0023 можуть використовуватися як восьмирівневий стек шістнадцятирозрядних слів або як комірки ОЗП даних з довільним доступом.

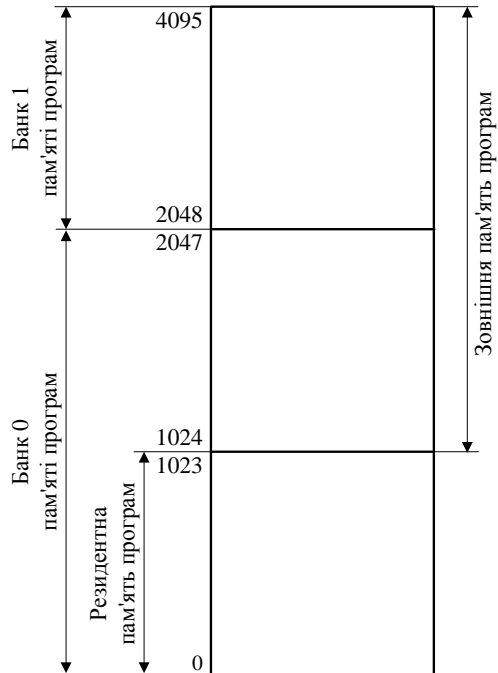


Рис. 3.4. Організація пам'яті програм

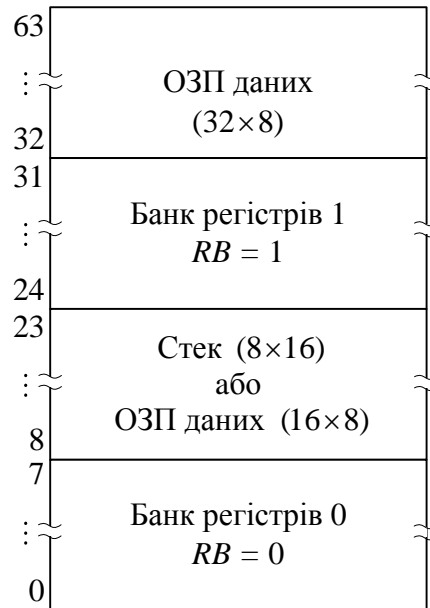


Рис. 3.5. Карта розподілу адрес внутрішньої пам'яті даних

Показчик стеку SP входить до складу регістра PSW (розряди $S2 - S0$). При переході до підпрограми в стек записується вміст PC і чотирьох старших розрядів PSW (рис. 3.6). Частина внутрішньої пам'яті з адресами $0032 - 0063$ використовується тільки в якості ОЗП даних.

Способи адресації для доступу до внутрішньої пам'яті даних:

- пряма регістрова;
- непряма регістрова.

Пряма регістрова адресація використовується для звернення до регістрів загального призначення $R0 - R7$ банку регістрів 0, якщо $RB = 0$, та $R0 - R7$ банку регістрів 1, якщо $RB = 1$. В цьому випадку трирозрядна адреса регістра присутня в коді команди.

За допомогою непрямої регістрової адресації можна звернутися до будь-якого байту внутрішньої пам'яті даних (в тому числі до регістрових банків і стеку). В якості показчика адресу операції в такому випадку використовуються регістри $R0$ і $R1$, відповідно до обраного банку регістрів (нульового або першого). Для обміну даними застосовуються наступні команди $MOV A, @Rr$, $MOV @Rr, A$, (де $r = 1, 0$).



Рис. 3.6. Організація стеку

Непряма регістрова адресація застосовується також під час виконання команд звернення до зовнішньої пам'яті даних. Виходячи з того, що в якості показчика адреси використовуються восьмирозрядні регістри $R0$, $R1$ обраного банку регістрів, максимальний об'єм зовнішньої пам'яті даних складає 256 байт. Під час обміну з зовніш-

ньою пам'яттю даних застосовуються команди `MOVX A, @Rr`, `MOVX @Rr, A`, (де $r = 1, 0$).

Стек

Внутрішній восьмирівневий стек забезпечує автоматичне збереження і відновлення основних параметрів обчислювального процесу при запитах переривання і при поверненні після обслуговування переривання.

Система переривань

В МК реалізована *система переривань* від двох джерел: внутрішнього таймера/лічильника і зовнішніх джерел переривань, наприклад від контролеру пріоритетних переривань.

Таймер/лічильник

Таймер/лічильник TCNT являється восьмирозрядним лічильником, що підсумовує, який можна читати і завантажувати через акумулятор *A*, використовуючи відповідні команди `MOV`. Він може працювати в режимі таймера і в режимі лічильника. В режимі таймера на вхід *TCNT* через дільники частоти поступають сигнали з частотою $F/480$, де F – частота, що задається кварцовим резонатором або зовнішнім генератором. Наприклад, при $F = 6$ МГц лічильник збільшує свій стан на 1 через кожні 80 мкс. Шляхом встановлення лічильника у певний вихідний стан і аналіз його переповнення можуть бути реалізовані різні часові затримки. Якщо 256 станів не забезпечують бажану затримку, то можна розрахувати декілька періодів роботи *TCNT*, накопичуючи в робочому регістрі необхідне число переповнень лічильника. Під час переходу *TCNT* із стану 255 в стан 0 ознака *TF* встановлюється в одиницю. Ця ознака може бути проаналізована командою `JTF`. Крім того, якщо переривання від *TCNT* дозволено командою `EN TCNT`, то при встановленні *TF* в 1 здійснюється перехід до підпрограми обслуговування переривання за вектором 0007. Переривання від *TCNT* може бути не дозволено командою `DIS TCNTI`. Після виконання команди `JTF` і при переході до підпрограми обслуговування переривання *TF* переходить в 0.

В режимі лічильника подій *TCNT* збільшує свій стан на 1 кожен раз, коли сигнал на вході *T1* переходить із стану 1 в стан 0. В режимі таймера *TCNT* запускається командою `STR T`, а в режимі лічиль-

ника – командою *STRT CNT*. Зупинка *TCNT* здійснюється командою *STOP TCNT* або системним скиданням.

Регістр слова стану програми

Крім описаних ознак *MB*, *RB*, і *TF* є також внутрішні ознаки *C*, *AC* і *F0*, які зберігаються в *PSW*, і ознака *F1*, яка до складу *PSW* не входить (не запам'ятовується у стеку при переході до підпрограми). Ознака *C* встановлюється у відповідності з переносом із сьомого розряду, яка формується при виконанні команд суми слів в арифметично-логічному пристрої. Вона встановлюється також при виконанні команд зсуву вмісту *A* в поєднанні з бітом *C*. Ознака *AC* встановлюється при виконанні команд суми, в залежності від значення переносу із третього розряду (переносу між тетрадами). Значення *AC* використовується командою *DA*, яка призначена для корекції вмісту акумулятора при сумуванні двійково-десятичних чисел. Ознаки *F0* і *F1* являються ознаками користувача. Існують і зовнішні ознаки *T0*, *T1* і *INT*, які формуються поза *IC*.

Під час виконання команд умовного переходу перевіряються ознаки *Z*, *Bb*, *C*, *F0*, *F1*, *T0*, *T1*, *TF* і *INT*. Виводи *T0*, *T1* і *INT* можуть використовуватися з іншою метою: *T0* – в якості видачі на об'єкт управління зовнішніх сигналів синхронізації з частотою $F/3$ (за командою *ENTO CLK*), *T1* – в якості входу таймера/лічильника (*STRT CNT*), *INT* – в якості входу зовнішнього переривання (*ENI*). Ряд ознак, а саме: *C*, *F0*, *F1*, *RB* і *MB* можна встановлювати програмно.

Порти вводу/виводу

Мікроконтролер вміщує три порти вводу/виводу: *P1*, *P2* і *BUS*. Порти *P1* і *P2* називають «квазідвоспрямованими». Їх особливість полягає в тому, що при вводі даних, над ними та поточним станом порту (даними, які виводилися із порту останніми) виконується порозрядна логічна операція *I*. Вихідні дані в порту запам'ятовуються. При скиданні системи кожному розряду порту присвоюється значення 1. У системі команд МК48 є команди, які дозволяють виконувати запис нулів і одиниць в будь-якому розряді або групі розрядів порту, але оскільки в цих командах маска задається безпосереднім операндом, то необхідно знати розподіл ліній, що скидаються і встановлюваних, на етапі розробки прикладної програми. В тому випад-

ку, якщо маска обчислюється програмою і наперед не відома, в ОЗП необхідно мати копію стану порту виведення. Ця копія по командах логічних операцій об'єднується з обчислюваною маскою в акумуляторі і потім завантажується в порт. Необхідність цієї процедури викликана тим, що в МК відсутня можливість виконати операцію читання значень портів $P1$ і $P2$ для визначення колишнього стану порту виведення. Порт $P2$ відрізняється від порту $P1$ тим, що його молодші чотири біта можуть бути використані для підключення додаткових портів $P4$, $P5$, $P6$, $P7$. Робота цих зовнішніх портів синхронізується сигналом PROG.

Порт BUS має звичайні двоспрямовані виходи з трьома станами. Порт застосовується для побайтового вводу/виводу даних. За допомогою команд ORL і ANL можливо маскувати байти, що передаються через порт, з ціллю обробляти у байті окремі біти або групу бітів. Команди звернення до портів включають безпосередній номер порту. У мікропроцесорних системах простої конфігурації, коли порт BUS не використовується як порт-розширювач системи, обмін виконується по командах INS, OUTU і MOVX. Можливе поперемінне використання команд OUTL і MOVX. Проте при цьому необхідно пам'ятати, що байт, який виводиться по команді OUTL фіксується в буферному регістрі порту BUS , а команда MOVX знищує вміст буферного регістра порту BUS . Команда INS не знищує вміст буферного регістра порту. В МПС що мають зовнішню пам'ять програм, порт BUS використовується для видачі адреси зовнішній пам'яті і для прийому команди із зовнішньої пам'яті програм. Отже, в таких системах використання команди OUTL позбавлене сенсу.

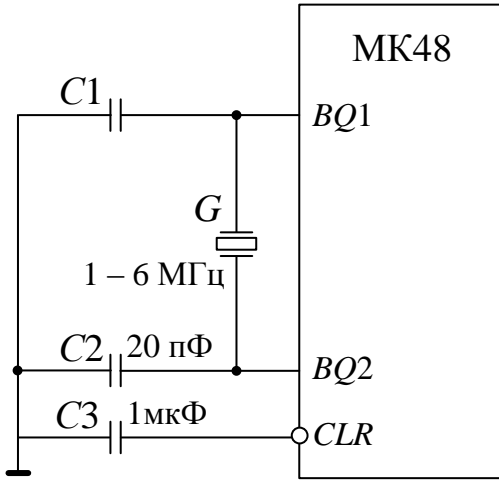
3.3. Основні режими роботи МК48

3.4.1 Ініціалізація системи

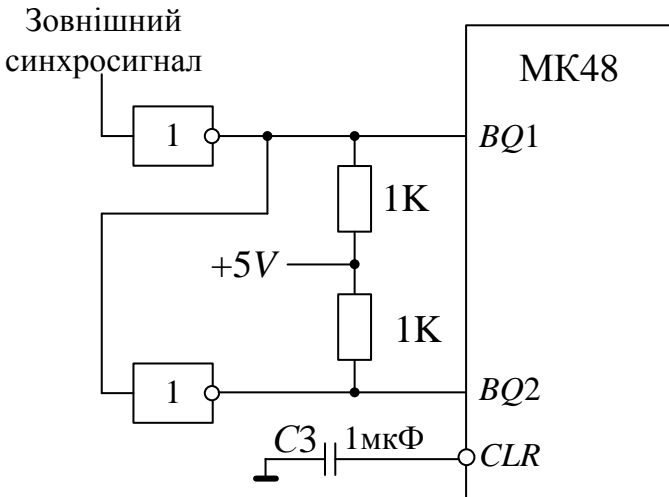
Частота синхронізації МК48 може бути задана за допомогою кварцового резонатора, схема включення якого вказана на рис. 3.7 а. Для синхронізації роботи МК48 з іншим обладнанням може також бути використаний зовнішній генератор (рис. 3.7. б).

Ініціалізація системи здійснюється за допомогою подачі на вхід CLR нульового сигналу, тривалістю не менш 50 мс. Ініціалізація системи може здійснюватись безпосередньо після включення джерела

живлення +5В при використанні схеми скидання (рис. 3.8 а), яка в найпростішому випадку складається із конденсатора $C3$.



а



б

Рис. 3.7. Схеми синхронізації і скидання мікро контролера: а – підключення кварцового резонатора; б – підключення зовнішнього генератора

В процесі ініціалізації забороняються переривання, порти $P1$ та $P2$ налаштовуються на ввід інформації, виходи порту BUS переводяться в високоомні стани (за встановлення $EMA = 0$), зупиняється таймер/лічильник $TCNT$, забороняється видача тактуючих сигналів на виході $T0$. Встановлюються в нуль тригери MB , TF , $F1$, $F0$, RB , а також регістри SP та PC . В результаті цього спочатку вибираються нульові банки пам'яті програм і регістрів, покажчик стеку SP вказує на нульову комірку стеку (байти ЗПД з адресами 0008 і 0009) та виконання програми починається з нульової адреси.

3.4.2. Режими роботи МК48 з пам'яттю

Режим роботи з резидентною пам'яттю програм

У режимі роботи МК48 з резидентною пам'яттю програм (для ВІС BE48 і ВІС BE49 за $EMA = 0$) для зв'язку з об'єктом управління використовуються три порти ($P1$, $P2$, BUS). При цьому кожний розряд будь-якого порту може бути запрограмований на ввід чи вивід інформації. За необхідності між виходами МК48 і входами/виходами об'єкту управління включаються відповідні елементи, що забезпечують формування сигналів за різними параметрами. Під час звернення до резидентної пам'яті програм зовнішні управляючі сигнали, окрім сигналу ALE , не формуються.

Сигнал ALE за необхідності можна використовувати для синхронізації роботи внутрішніх пристроїв. Якщо адреса перевищує допустиме для резидентної пам'яті програм значення – 1023, то МК48 автоматично переходить в режим роботи з зовнішньою пам'яттю програм.

Режим роботи з зовнішньою пам'яттю програм

Режим роботи МК48 з зовнішньою пам'яттю програм можливий за застосування додаткових мікросхем ПЗП. Якщо не використовувати сторінкову адресацію, об'єм пам'яті програм можна розширити до 4К байт. За встановлення сигналу $EMA = 1$ доступні всі 4К байта зовнішньої пам'яті. Якщо сигнал $EMA = 0$, то адресація комірок зовнішньої пам'яті розпочинаються з адреси 1024. При цьому область пам'яті з адресами від 0 до 1023 належать резидентній пам'яті програм. Схема підключення зовнішньої пам'яті програм об'ємом 4К наведена на рис. 3.8, а часова діаграма читання байта (команди або даних) – на рис. 3.9.

Для підключення зовнішньої пам'яті програм використовуються виходи портів $BUS[7..0]$ та $P2[3..0]$. Для зберігання адреси звернення до пам'яті використовується зовнішній регістр адреси PA .

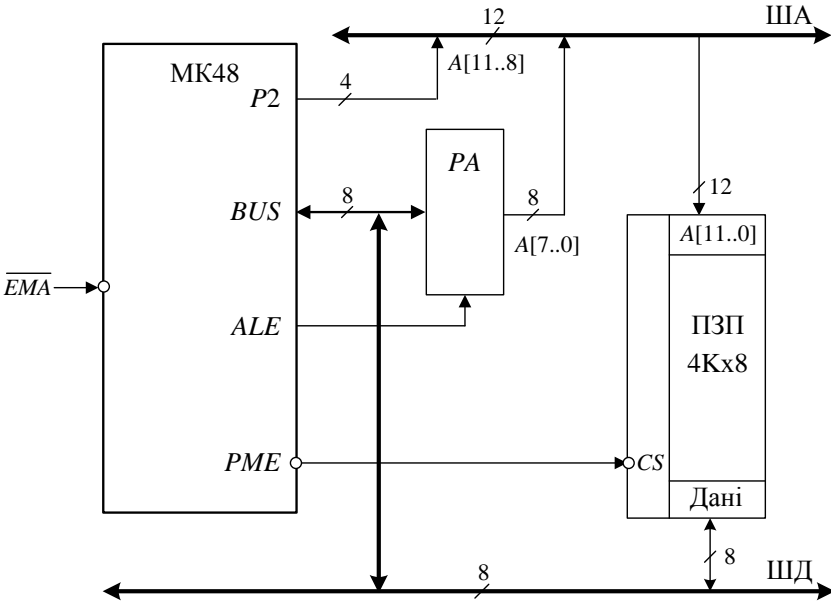


Рис. 3.8. Схема підключення зовнішньої пам'яті програм

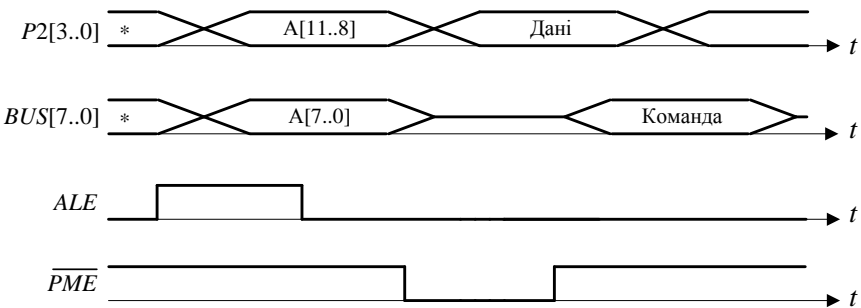


Рис. 3.9. Часова діаграма вибору інформації з зовнішньої пам'яті програм

В процесі звернення до зовнішньої пам'яті програм МК48 формує дванадцятирозрядну адресу, де старші розряди адреси $A[11..8]$

формується на виходах $P2[3..0]$ мікросхеми, а молодші розряди $A[7..0]$ – на виходах $BUS[7..0]$. Запис адреси в реєстр адреси PA стробується сигналом ALE . Після цього виходи порту BUS перемикаються на ввід інформації та видається сигнал дозволу читання пам'яті PME . За цим сигналом на виходах даних мікросхеми пам'яті формується байт інформації (команди або даних), який за шиною BUS приймається в МК48.

За випадку, коли резидентна пам'ять програм відсутня ($EMA = 0$), необхідно передбачити логіку відключення резидентної пам'яті під час звернення до адрес $0 - 1023$. Наприклад, якщо вибір зовнішньої пам'яті здійснюється низьким рівнем сигналу на вході CS , то на цей вхід можна подати значення логічної функції $NOT(P2[3] OR P2[2])$.

Режим роботи з зовнішньою пам'яттю даних

В режимі роботи МК48 з зовнішньою пам'яттю даних використовуються додаткові мікросхеми ОЗП об'ємом 256 байт.

Якщо адресний простір має об'єм більш ніж 256 байт, то необхідна сторінкова організація зовнішньої пам'яті даних. При роботі з адресами в межах однієї сторінки застосовуються команди $MOVX A, @Rr$, $MOVX @Rr, A$ (де $r = 1, 0$). Обмін інформацією здійснюється між акумулятором A і коміркою ОЗП, яка непрямо адресується через реєстр $R0$ або $R1$. Переключення між сторінками потребує використання додаткових команд вибору сторінки пам'яті даних.

На рис. 3.10 показаний спосіб підключення до МК48 n сторінок зовнішньої пам'яті даних з використанням k виходів порту $P1$ (де $k = \lceil \log_2 n \rceil$) і дешифратора DC .

За наявності одночасно зовнішньої пам'яті програм і даних використовується один і той самий зовнішній реєстр адреси.

Часові діаграми циклів звернення до зовнішньої пам'яті даних наведені на рис. 3.11.

Команди виконуються за два цикли. В процесі читання пам'яті спочатку на шину BUS встановлюється адреса, яка фіксується у зовнішньому реєстрі PA за встановлення сигналу ALE . Після чого виходи порту BUS перемикаються для вводу інформації і формується сигнал читання даних R . За цим сигналом ЗПД виставляє дані на ШД, які через порт BUS приймаються в МК48. На протязі циклу запису інформація, що передається, встановлюється на шині BUS . Запис в пам'ять стробується сигналом W .

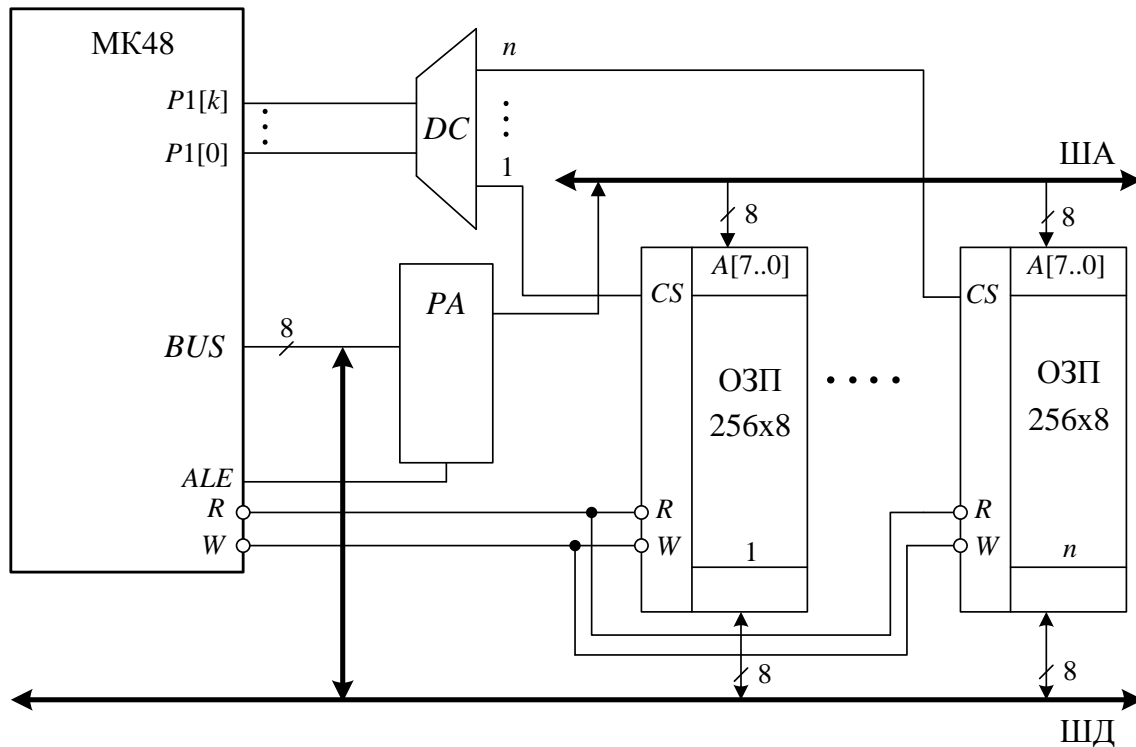


Рис. 3.10. Схема підключення зовнішньої пам'яті даних

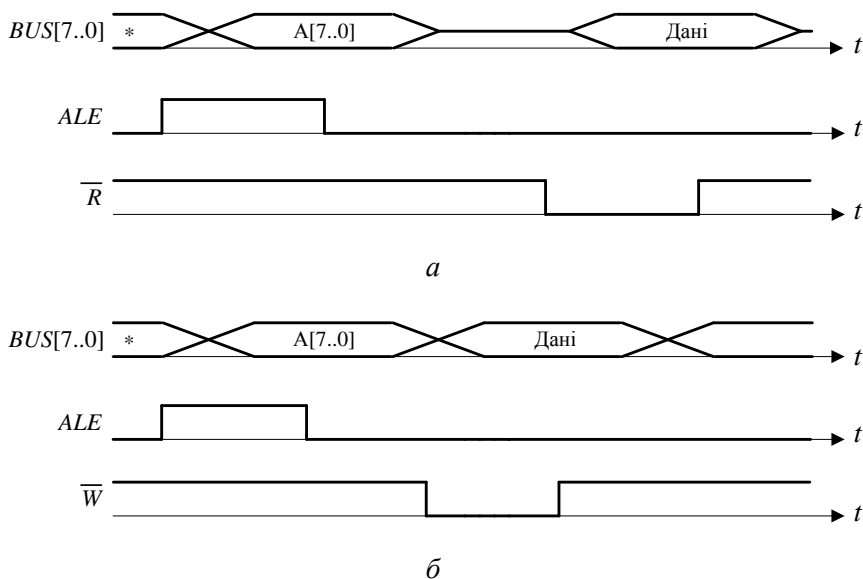


Рис.3.11. Часові діаграми обміну даними з зовнішньою пам'яттю даних: *а* – читання даних; *б* – та запису даних

Обмін інформацією між МК48 та зовнішньою пам'яттю (як з пам'яттю програми, так і з пам'яттю даних) здійснюється в синхронному режимі, тобто на обмін виділяється фіксований проміжок часу, причому сигнали квитирування (зворотного зв'язку), узгодженні за часом роботи МК48 і пам'яті, не передбачаються. В зв'язку з цим таке узгодження можливо реалізувати тільки за рахунок вибору частоти тактування роботи МК48.

Для реалізації більш складних програмно-апаратних способів доступу до зовнішньої пам'яті, її ємність може бути збільшена до необхідного об'єму за рахунок сторінкової організації, при цьому зовнішня пам'ять даних поділяється на сторінки по 256 байт в кожній, а зовнішня пам'ять програм – на сторінки по 4К байт. Для переключення між сторінками можна використовувати, вільні лінії портів $P1$ та $P2$.

Режим покрокового виконання програми

Режим покрокового виконання програми застосовується при налагодженні мікропроцесорних систем. Для його реалізації використовується вхід мікросхеми МК48 SS (рис. 3.1). Під час надходження

на цей вхід сигналу низького рівня МК48 завершує виконання чергової команди і зупиняється на етапі вибірки наступної команди. Підтвердженням зупинки є високий рівень сигналу *ALE*. В цьому стані на виходах МК48 *P2[3..0]* та *BUS* вже встановлена адреса наступної команди, яка може бути використана при аналізі стану системи в процесі налагодження. Для виходу МК48 з режиму зупинки необхідно подати сигнал високого рівня на вхід *SS*. При цьому на виході *ALE* встановлюється сигнал низького рівня і МК48 виконує чергову команду.

Схема реалізації режиму покрокового виконання програми ілюструється на рис. 3.12. Для вибору покрокового режиму використовується перемикач «Покроковий-автоматичний». *RS*-тригер застосовується для **заглушення брязкоту контактів** під час натискання кнопки «Крок». За натискання кнопки «Крок» запускається покроковий режим, при цьому *D*-тригер встановлюється в одиницю, тобто на вхід *SS* мікросхеми поступає додатний потенціал. Мікроконтролер завершує виконання чергової команди, виставляє адресу наступної команди і формує низький рівень сигналу *ALE*. В результаті цього *D*-тригер встановлюється в нульовий стан, що приводить до надходження низького рівня сигналу на вхід *SS*. Мікроконтролер переходить в режим зупинки до чергового натискання кнопки «Крок».

Часова діаграма сигналів при роботі МК48 в покроковому режимі показана на рис. 3.13.

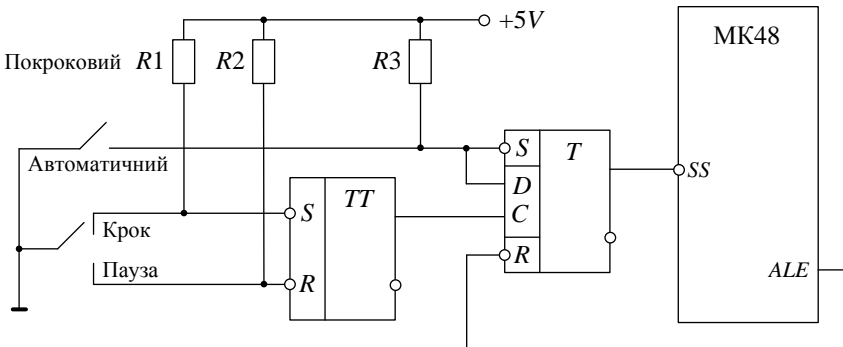


Рис. 3.12. Схема завдання режимів роботи мікроконтролера

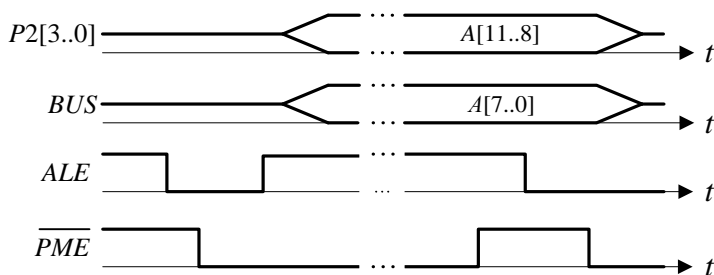


Рис. 3.13. Діаграма покрокового виконання команд

3.4.3. Підключення додаткових портів

Для збільшення кількості ліній зв'язку МК48 з об'єктом управління підключають додаткові чотирирозрядні порти $P4$, $P5$, $P6$, $P7$. Найбільш просто це здійснюється за використання спеціальної ІС KP580 BP43, спосіб підключення якої до МК48 показаний на рис. 3.14. В цьому випадку забезпечується виконання всіх чотирьох команд роботи з додатковими портами – $\text{MOVD } A, Pp$; $\text{MOVD } Pp, A$; $\text{ANLD } Pp, A$ та $\text{ORLD } Pp, A$ (де $p = \overline{4, 7}$), причому кожний вихід порту може бути налаштований як на введення так і на виведення інформації.

Команди передачі інформації між МК48 та додатковими портами виконуються за два цикли. В першому циклі на виходах $P2[3..0]$ встановлюється управляюче слово, в другому циклі – через зазначені виходи здійснюється обмін інформацією між МК48 та одним з додаткових портів. Формат управляючого слова показаний на рис. 3.15. Для стробування даних в режимі підключення додаткових портів використовується сигнал $PROG$.

Відмітимо, що логічні операції І та АБО виконуються безпосередньо в ІС BP43. Це необхідно враховувати при побудові додаткових портів з використанням інших апаратних засобів.

Часова діаграма роботи з додатковими портами $P4$, $P5$, $P6$, $P7$ показана на рис. 3.16.

Для розширення функціональних можливостей системи до МК48 можна підключати різні ІС, наприклад, адаптери KP580BB51, KP580BB55 тощо.

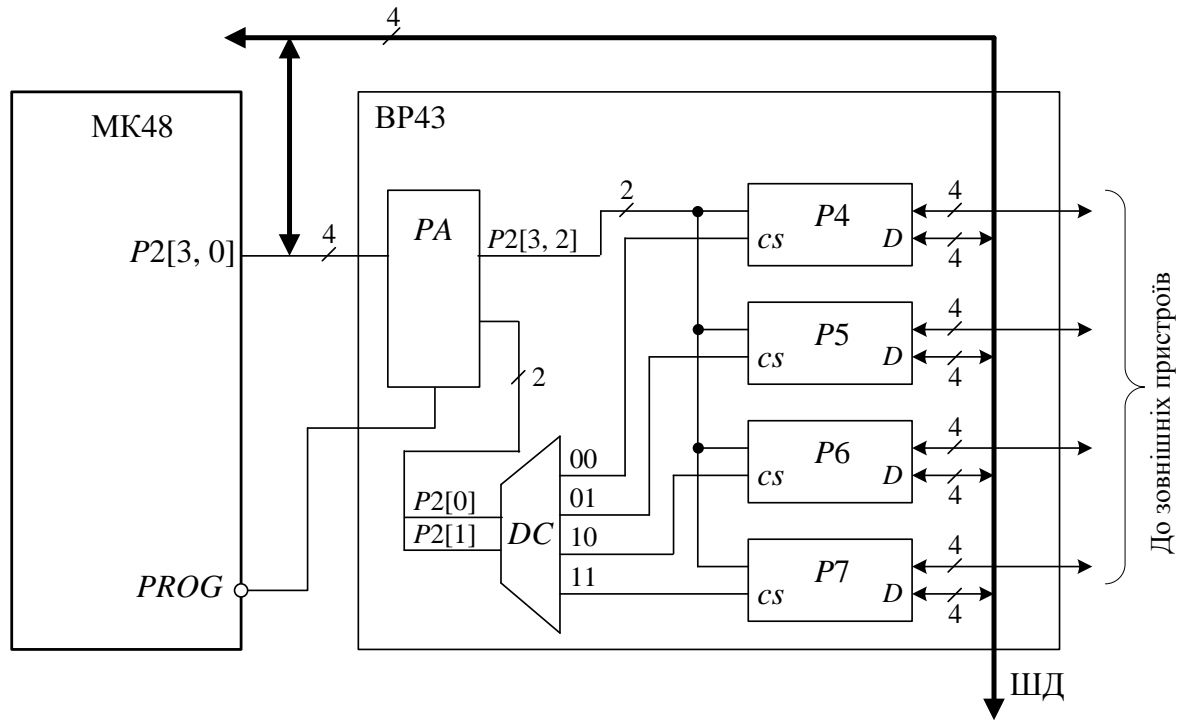


Рис. 3.14. Схема з'єднання виходів МК та ІС КР580ВР43

	3	2	1	0	
	↓	↓	↓	↓	
MOVD A, Pp	0	0	0	0	P4
MOVD Pp, A	0	1	0	1	P5
ANLD Pp, A	1	0	1	0	P6
ORLD Pp, A	1	1	1	1	P7

Рис. 3.15. Структура управляючого слова

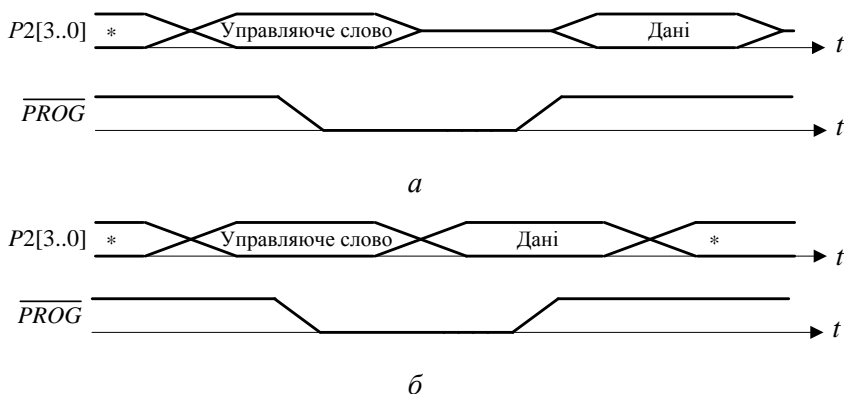


Рис. 3.16. Часова діаграма роботи з додатковими портами P4, P5, P6, P7: а- ввід даних, б- вивід даних

3.4.4 Підключення програмованого периферійного адаптера K580BB55

Основне призначення програмованих периферійних адаптерів (ППА) розробка програмованих пристроїв вводу/виводу для МПС.

Програмований периферійний адаптер 580BB55 (закордонний аналог IC 8255A) виготовляється за *NMOS* технології. Може бути застосований у МПС з мікропроцесорами МК51, МК48, МП8086.

На рис. 3.17 показано умовне графічне позначення ППА K580BB55.

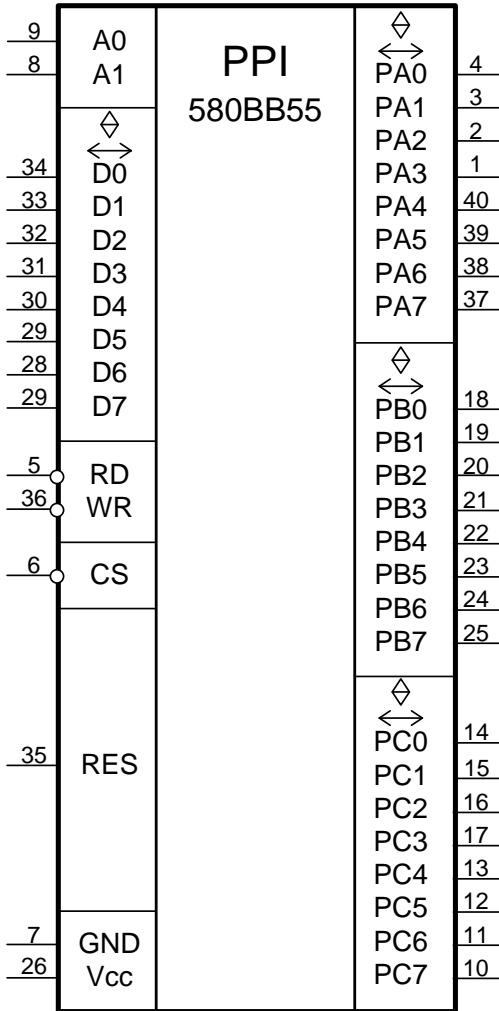


Рис. 3.17. Умовне графічне позначення програмованого периферійного адаптера K580BB55

Основні призначення виходів мікросхеми K580BB55:

- A0, A1 – сигнали двох молодших розрядів шини адреси
- CS – вибір кристалу
- RD – сигнал читання даних із ППА
- WR – сигнал запису даних
- RESET – сигнал установки початкового стану
- D7 – D0 – сигнали розрядів шини даних МК48

$PA7 - PA0$ – входи/виходи порту PA
 $PC7 - PC0$ – входи/виходи порту PC
 $PB7 - PB0$ – входи/виходи порту PB
 GND – загальний;
 Vcc – напруга живлення +5В;

Структурна схема програмованого периферійного адаптера K580BB55 зображена на рис. 3.18.

Адаптер 580BB55 забезпечує ввід/вивід за трьома додатковими восьмирозрядними портами PA , PB , PC . Причому порт PC може застосовуватися в якості двох чотири розрядних портів PCh – старша тетрада порту PC , та PCl – молодша тетрада порту PC .

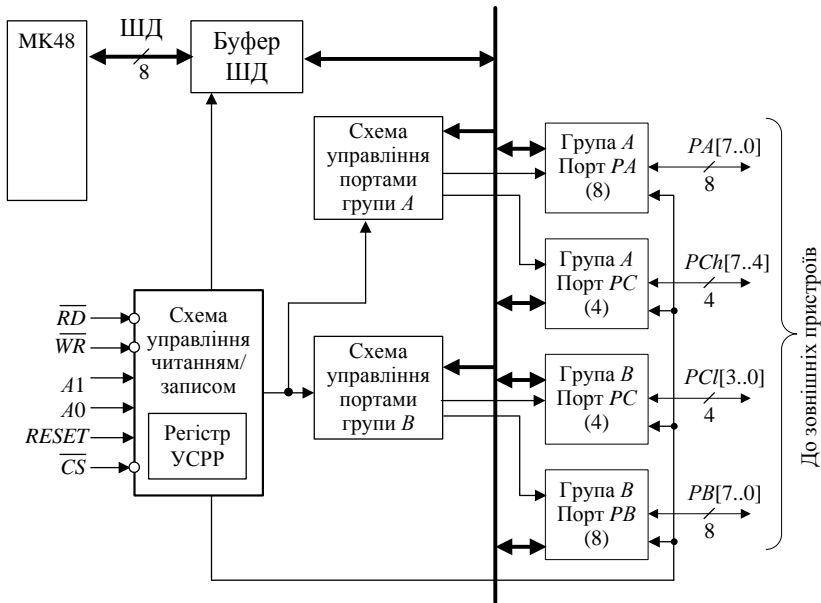


Рис. 3.18. Структурна схема програмованого периферійного адаптера K580BB55

До складу ППА входять наступні функціональні блоки.

- буфер шини даних (ШД) $D7 - D0$;
- схема управління читанням/записом даних в регістри ППА;
- група А, порт PA – порт вводу/виводу PA групи А;
- група В, порт PB – порт вводу/виводу PB групи В;
- група С, порт PC – порт вводу/виводу PCh групи А;
- група В, порт PC – порт вводу/виводу PCl групи В;

- схема управління портами групи А: порти PA та PCh ;
- схема управління портами групи А: порти PB та PCI .

Схеми управління портами групи А та В містять регістр управління, що задає режими роботи портів.

Всі порти оснащені буферними регістрами, через які здійснюється зв'язок між ППА і зовнішніми шинами.

Програмування режимів роботи і управління ІС здійснюється мікропроцесором за допомогою сигналів $D7 - D0$, $A1$, $A0$, \overline{CS} , \overline{RD} , \overline{WR} , $RESET$.

Адреси портів ППА входять до загального адресного простору зовнішньої пам'яті даних. Доступ до портів під час запису та читання здійснюється за застосування команд $MOVX A, @Rr$; $MOVX @Rr, A$ (де, $r = 1, 0$).

Відмітимо, що по шині даних відбувається не тільки обмін даними, але і пересилання з МК48 в ППА управляючих слів, генерованих програмним забезпеченням процесора, а також передача в МК48 інформації про стан периферійного обладнання. Низький рівень сигналу на вході вибору кристалу CS дозволяє інформаційний зв'язок між ППА і МП48.

Обмін між МК48 і регістрами портів PA , PB і PC організовується під управлінням сигналів, що подаються на входи схеми управління читанням/записом. Функціональне призначення портів PA , PB і PC визначається управляючим словом, яке завантажується процесором у регістр управляючого слова режиму роботи УСРР (рис. 3.18). Сигнали на адресних входах $A0$ і $A1$ мікросхеми K580BB55 виробляють селекцію одного з трьох портів PA , PB , PC або регістру УСРР.

Восьмирозрядні порти програмованого периферійного адаптера можуть бути використані різними способами в залежності від характеристик конкретного устаткування.

Основні операції, які реалізуються ППА надані в табл. 3.2.

Управляюче слово режиму роботи (УСРР) використовується для налаштування режимів роботи ППА.

Програмуючий периферійний адаптер може знаходитися в таких основних режимах:

- «0» – основний режим вводу/виводу інформації;
- «1» – режим стробуючого вводу/виводу інформації;
- «2» – режим двоспрямованої шини.

Таблиця 3.2. Основні операції, що виконує програмований периферійний адаптер K580BB55

Дії	Сигнали управління					Операції
	A1	A0	\overline{WR}	\overline{RD}	\overline{CS}	
Читання	0	0	0	1	0	РА ← ШД (вивід в порт PA)
	0	1	0	1	0	РВ ← ШД (вивід в порт PB)
	1	0	0	1	0	РС ← ШД (вивід в порт PC)
Запис	0	0	1	0	0	ШД → РА
	0	1	1	0	0	ШД → РВ
	1	0	1	0	0	ШД → РС
	1	1	1	0	0	ШД → РУС
Відключення	*	*	*	*	*	ШД та порти відключені

Формат УСРР наведений на рис. 3.19.

Будь-який із перерахованих режимів може бути вибраний в ході виконання системної програми і встановлений завантаженням управляючого слова в регістр УСРР. Це дозволяє простими програмними засобами здійснювати реконфігурацію периферії МК48. Під час запису УСРР всі регістри пам'яті портів встановлюються в нуль, розряд D[7] – ознака встановлення режиму роботи, повинен бути встановлений у одиницю.

Якщо розряд D[7] = 0, управляюче слово не записується в регістр УСРР (на виходах D7 – D0 формується управляюче слово маніпуляції з бітами – УСМБ), в цьому випадку відбувається запис в один з тригерів регістру пам'яті порту РС значення 0 або 1, визначене розрядом D[0]. Розряди D[3] – D[1] задають номер тригера в регістрі пам'яті порту РС, в який завантажуються значення вказане в розряді D[0] – встановлення/скидання біту порту РС. Формат УСМБ зображено на рис. 3.20.

Порт РС в цьому випадку є регістром пам'яті з розрядами, що адресуються. Розряд D[7] в управляючому слові, що надходить з шини даних адресує два внутрішніх регістра: регістр УСРР і регістр пам'яті порту РС. Такий спосіб передачі адресних сигналів внутрішніх вузлів по шині даних сприяє зменшенню числа зовнішніх виводів ІС.

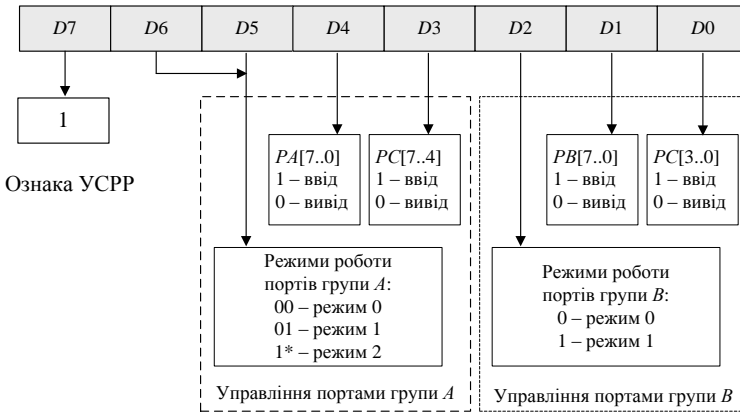


Рис. 3.19. Формат управляючого слова режиму роботи

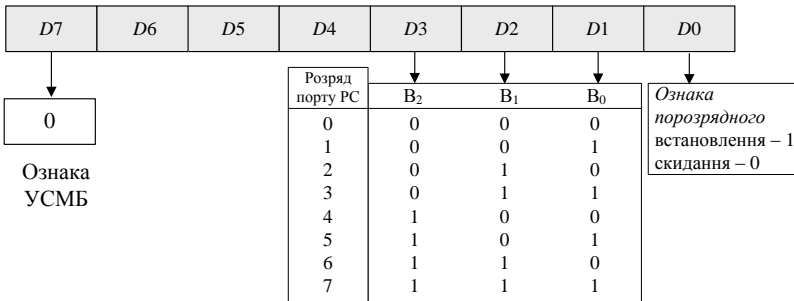


Рис. 3.20. Формат управляючого слова маніпуляції з бітами

Любий із розрядів порту PC може бути вибраний в якості тригера запиту переривання ЗПР або тригера фіксації дозволу переривання РПР. Програмний доступ до розрядів порту PC дає можливість розробити різноманітні процедури обробки переривань, пристосувавши їх до структури мікропроцесорної системи.

На рис. 3.21 приведена схема підключення програмованого периферійного адаптера до мікроконтролера МК48. За наявності одночасно зовнішньої пам'яті програм і зовнішньої пам'яті даних використовується один загальний зовнішній регістр адреси PA . Адреси портів PA , PB , PC та регістра управляючого слова РУС програмованого периферійного адаптера входять в загальний адресний простір однієї сторінки зовнішньої пам'яті даних. Для вибору адреси ППА застосовується селектор адреси SA .

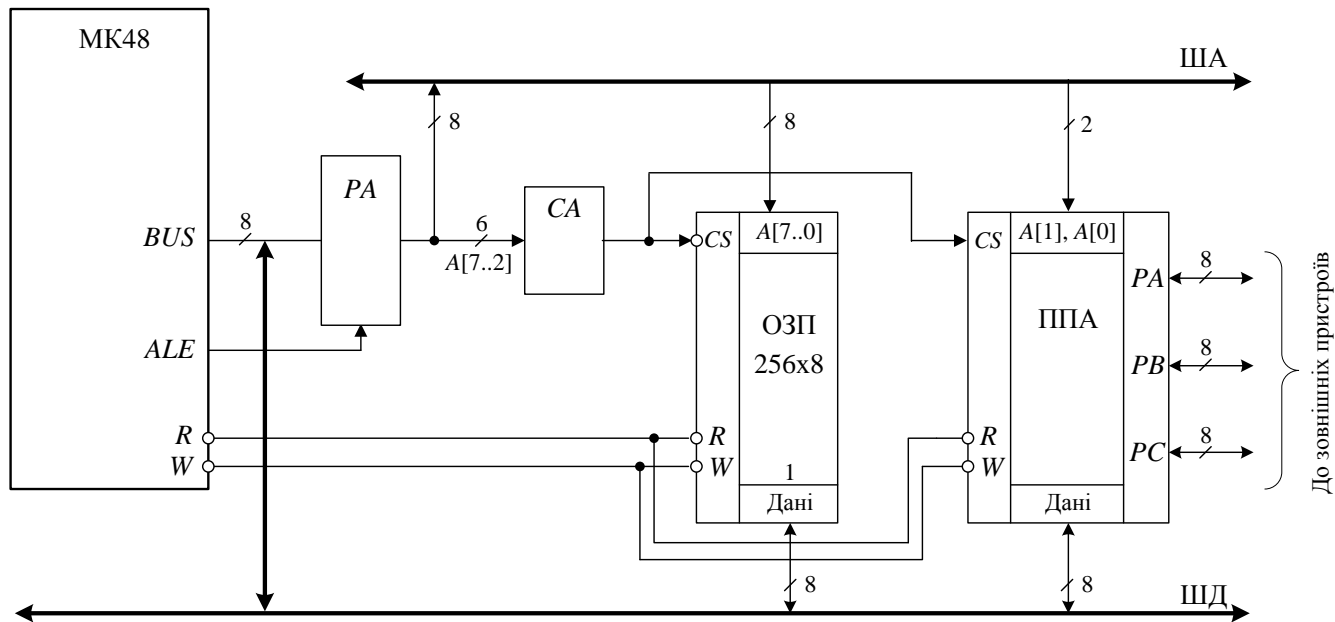


Рис. 3.21. Схема підключення програмованого периферійного адаптера K580BB55 до мікроконтролера МК48

3.4.5. Підключення програмованого зв'язувального адаптера K580BB51

В мікропроцесорних системах передача даних між функціональними пристроями здійснюється через системну магістраль. Дані передаються байтами в паралельному форматі. Передача даних в зовнішніх каналах зв'язку відбувається у вигляді послідовного потоку розрядів – в послідовному форматі. Для організації послідовних каналів зв'язку у МПС застосовуються додаткові пристрої – адаптери. Адаптери перетворюють дані, які знімаються з ШД з паралельного формату у послідовний формат, що придатний для передачі в відповідний канал зв'язку; а дані в послідовному форматі, що приймаються з каналу зв'язку, перетворює в паралельний формат, придатну для прийому в МПС.

Програмовані зв'язувальні адаптери ПЗА є універсальними приймачами/передавачами, призначеними для роботи в ланцюгах послідовного синхронного/асинхронного обміну.

Програмований зв'язувальний адаптер реалізований у вигляді ІС K580BB51 (закордонний аналог ІС 8251, та удосконалена ІС 8251А), умовне графічне позначення якого зображене на рис. 3.22.

Виготовляється ІС за *n*-МОП технології, входи/виходи сумісні з TTL ІС.

Основні призначення виходів мікросхеми K580BB51:

<i>D7 – D0</i>	– сигнали шини даних;
<i>RESET</i>	– сигнал установки початкового стану, очікування УСРР;
<i>CLK</i>	– сигнал синхронізації;
<i>C/D</i>	– управління/дані, адресний розряд <i>A0</i> (див. табл. 3.5): 0 – читання/запис даних, 1 – читання/запис управляючих слів або слова стану;
\overline{RD}	– читання;
\overline{WR}	– запис;
\overline{CS}	– вибір мікросхеми, сигнал з дешифратора адресних розрядів <i>A[7..1]</i> ;
$T \times D$	– вихідні послідовні дані передавача;
$T \times C$	– тактовий сигнал передавача, частота якого визначає швидкість передачі за послідовним каналом зв'язку (біт/сек);

$T \times RDY$	–	готовність передавача, вказує МК48 на готовність буфера передавача прийняти байт даних для передачі у послідовний канал зв'язку;
$T \times E$	–	завершення передачі ($T \times EMPTY$): 1 – відсутність у буфері даних передавача чергового символу для передачі за послідовним каналом зв'язку; 0 – отримання символу від МК48 при умові дозволу на передачу;
$R \times D$	–	вхідні послідовні дані приймача;
$R \times C$	–	тактовий сигнал приймача, частота якого визначає швидкість передачі за послідовним каналом зв'язку (біт/сек) (зазвичай використовується один і той самий генератор тактових імпульсів для приймача і передавача, так що $T \times C = R \times C$);
$R \times RDY$	–	готовність приймача;
$SYNDET$	–	$SYNDET / BRKDET$ (виявлення синхронізації/ виявлення паузи): в синхронному режимі прийому є вихідним сигналом ($SYNDET$) під час завдання режиму внутрішньої синхронізації, є вхідним сигналом ($SYNDET$) під час завдання режиму зовнішньої синхронізації; в асинхронному режимі прийому ($BRKDET$) є вихідним сигналом;
GND	–	загальний;
V_{cc}	–	напруга живлення +5В;
\overline{DTR}	–	запит готовності передавача терміналу ($\overline{DTR} = 0$), вихідний сигнал до модему застосовується для запиту його готовності передавати дані у ПЗА;
\overline{DSR}	–	готовність передавача терміналу ($\overline{DSR} = 0$), вхідний сигнал, що поступає від модему у відповідь на сигнал \overline{DTR} , вказує на готовність модему передати дані у ПЗА;
\overline{RTS}	–	запит готовності приймача терміналу ($\overline{RTS} = 0$); вихідний сигнал до модему застосовується для запиту його готовності прийняти дані від ПЗА;
\overline{CTS}	–	готовність приймача терміналу, вхідний сигнал, що поступає від модему у відповідь на сигнал \overline{RTS} , застосовується для дозволу передачі даних від у ПЗА у модем;

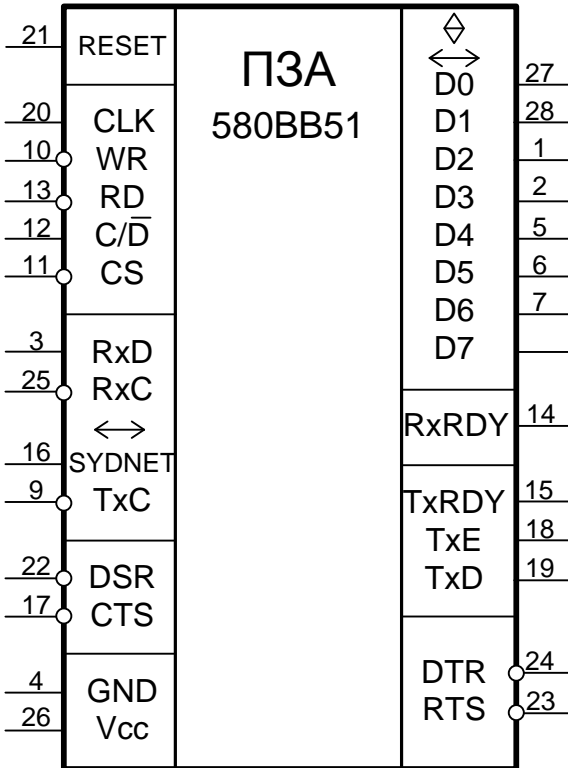


Рис. 3.22. Умовне графічне зображення програмованого зв'язувального адаптера K580BB51

Структурна схема ПЗА K580BB51 показана на рис.3.23.

До складу ПЗА входять:

- *буфер (приймач/передавач) шини даних (ШД)*, що є восьмирозрядним регістром з трьома станами й використовується для обміну даними та управляючими словами;

- *блок управління записом/читанням*, що містить восьмирозрядні регістри: регістр управляючого слова режиму роботи (УСРР), регістр управляючого слова команди (УСК) та регістр управляючого слова статусу (УСС); блок записом/читанням приймає сигнали від МК48 та генерує внутрішні управляючі сигнали;

- *буфер передавача (перетворення $P \rightarrow S$)*, який призначено для приймання символів у паралельному форматі P (*Parallel*) від МК48 та видачі послідовного потоку розрядів S (*Serial*) на вихід

$T \times D$; буфер передавача (рис. 3.24) містить регістр вводу даних від МК48 та регістр зсуву PI/SO (*Parallel Input/Serial Output*), який перетворює дані з паралельного формату на послідовний;

– *схема управління передавачем*, застосовується для управління передачею;

– *буфер приймача* (перетворення $S \rightarrow P$), який виконує прийом послідовного потоку даних з входу $R \times D$ та передачу їх в МК48 в паралельному форматі; буфер приймача (рис. 3.24) містить регістр виводу даних в МК48 та регістр зсуву SO/PI (*Serial Output/Parallel Input*), який перетворює дані з послідовного формату на паралельний;

– *схема управління приймачем* застосовується для управління передачею, автоматично фіксує виявлені помилки парності, переповнення, кадру в управляючому слові стану;

– *схема управління модемом* (модулятор/демодулятор), що обробляє управляючи сигнали, призначені для зовнішнього пристрою.

Схема управління модемом

Сигнали управління модемом застосовуються для квитирування передачі послідовних даних між ПЗА та модемом. **Квитирування – сигнал зворотного зв'язку, який призначений для інформування передавача про закінчення роботи приймача. Після чого передавач завершує операцію обміну та готовий до наступної передачі даних.**

Устаткування, що приймає участь у передачі даних за послідовними калами зв'язку поділяють на два типи: термінальне устаткування – комп'ютери принтери, сканери і таке інше та зв'язувальне. Адаптер ПЗА входить до класу термінального устаткування. Типовим прикладом зв'язувального устаткування є модем, що з'єднує, наприклад, комп'ютер та телефонну лінію. Під час передачі даних за послідовними лініями зв'язку на обох кінцях лінії працюють модеми, що перетворюють цифрові сигнали на аналогові із застосуванням того чи іншого способу модуляції, під час прийому відбувається зворотне перетворювання – демодуляція.

Структурна схема дуплексного каналу зв'язку зображена на рис. 3.25. Для управління прийомом та передачею даних між ПЗА та модемом призначені пари сигналів \overline{DTR} , \overline{DSR} та \overline{RTS} , \overline{CTS} .

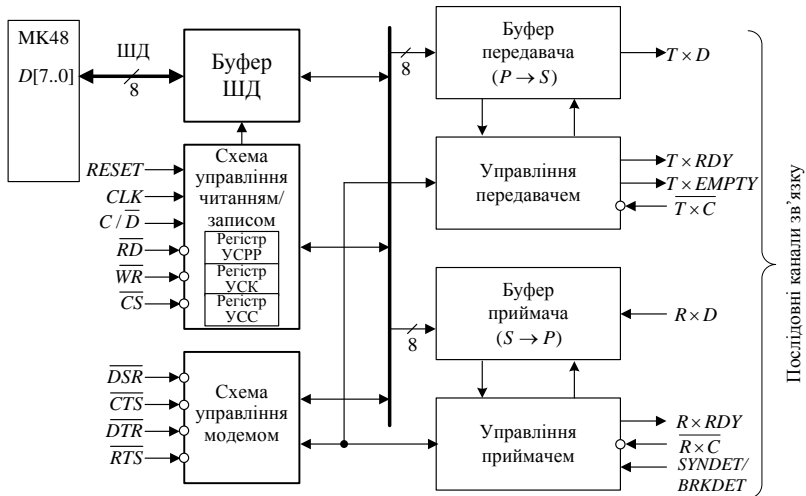


Рис. 3.23. Структурна схема програмованого зв'язувального адаптера K580BB51

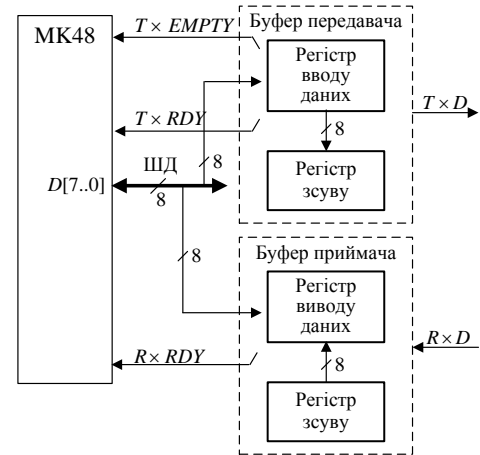


Рис. 3.24. Структурна схема буферів передавача та приймача

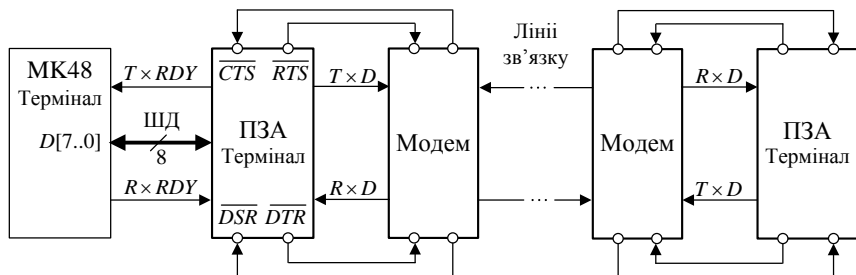


Рис. 3.25. Структурна дуплексного схема каналу зв'язку

Управління даними та режимами роботи ПЗА

Через буфер шини даних за сигналом запису \overline{WR} МК48 записує у блок управління записом/читанням ПЗА дані або управляючі слова режиму роботи та команди (УСРР та УСК). За сигналом читання \overline{RD} МК48 зчитує з блоку управління записом/читанням ПЗА дані або управляюче слово стану (УСС).

Сигнал на вході C/\overline{D} вказує на тип інформації, що він приймає: управляюче слово, якщо $C/\overline{D} = 1$, дані, якщо $C/\overline{D} = 0$. Будь які операції обміну даними можливі тільки в тому випадку, якщо на вході вибору мікросхеми \overline{CS} встановлений нульовий сигнал.

Основні режими роботи ПЗА приводяться в табл. 3.3. Основні сигнали управління (C/\overline{D} , \overline{RD} , \overline{WR} , \overline{CS}) подаються на схему управління читанням/записом від МК48 та визначають тип оброблюваної інформації та напрямок передачі. Режим роботи ПЗА задається програмним шляхом після завантаження в нього управляючих слів УСРР та УСК з МК48.

Таблиця 3.3. Основні режими роботи програмованого зв'язувального адаптера K580BB51

Вхідні сигнали				Операція (тип інформації)
C/\overline{D}	\overline{RD}	\overline{WR}	\overline{CS}	
0	0	1	0	ПЗА → ШД (ввід даних)
0	1	0	0	ШД → ПЗА (вивід даних)
1	0	1	0	ПЗА → ШД (читання УСС)
1	1	0	0	ШД → ПЗА (запис УСРР)
*	1	1	0	Немає операції (ШД у Z-стані)
*	*	*	1	Немає операції (ШД у Z-стані)

Таким чином, програмований зв'язувальний адаптер оперує з управляючими словами наступних типів:

- управляюче слово режиму роботи;
- управляюче слово команди;
- управляюче слово стану.

Управляюче слово режиму роботи задає синхронний або асинхронний режим роботи, формат даних, швидкість прийому або передачі, необхідність контролю. Дані в регістр УСРР заносяться одразу після установки ПЗА в початковий стан програмно або за сигналом «RESET» та замінюються тільки під час зміни режиму.

Управляюче слово команди здійснює управління встановленим режимом обміну та може багаторазово змінюватись в процесі обміну даними під час управління його етапами.

В *управляючому слові стану* під час прийому даних за каналом зв'язку ПЗА фіксує виявлені помилки парності, переповнення, **кадру**.

В *асинхронному режимі* обмін даними відбувається із заздалегідь визначеною швидкістю. Приймач повинен бути узгоджений з передавачем за всіма параметрами формату сигналу, що передається, в тому числі і за часом передачі одного символу. В асинхронному режимі не вимагається сигнал зворотного зв'язку (квитирування), який призначений для інформування передавача про закінчення роботи приймача.

Синхронний режим характеризується наявністю зворотного зв'язку приймача з передавачем, за яким передавач завершує операцію обміну та готовий до наступної передачі даних.

Під час асинхронного режиму управляюче слово команди завантажуються одразу після управляючого слова режиму роботи, а під час синхронного режиму, перед УСК розташовуються один або два символи синхронізації.

В асинхронному режимі роботи формат кадра даних що передається за послідовними каналами зв'язку включає нульовий старт-біт, біти даних, контрольний біт та стоп-біти (рис. 3.26).

Високий рівень напруги називається пропуском, низький – маркером. Кадр розпочинається з передачі старт-біта (пропуску), далі передаються розряди даних, розпочинаючи з молодшого розряду, біт паритету P (контрольний розряд для визначення помилки за допомогою контролю на парність, або непарність) та один, півтора або два стоп-біта (маркери). Управляюче слово режиму роботи задає кіль-

кість бітів даних, кількість стоп-бітів, наявність біта контролю, та режим контролю.



Рис. 3.26. Формат кадру даних під час асинхронної передачі даних

Приймач кадру даних закінчується значенням стоп-біта. Якщо зчитаний пропуск (0) то в УСС фіксується *помилка кадру*. Якщо визначений маркер (1), приймач перетворює прийнятий послідовний код символу у паралельний і інформує МК48 про готовність даних. Якщо черговий прийнятий ПЗА з каналу зв'язку символ, не буде вчасно прочитаний МК48, він заміщується новим прийнятим символом і фіксується *помилка переповнення*. Приймач здійснює перевірку прийнятого символу на парність або непарність. *Помилка паритету* також фіксується у УСС. Слід зазначити, що виявлення такої помилки не зупиняє роботу приймача.

Формат управляючого слова режиму роботи

Формат УСРР для асинхронного режиму обміну даними зображено на рис. 3.27.

Призначення розрядів управляючого слова режиму роботи для асинхронного режиму обміну даними наступне:

- розряди D_0, D_1 ($D_0, D_1 \neq 0$) – визначають три різновиди асинхронних режимів, що визначаються коефіцієнтом відношення частоти тактового сигналу передавача $T \times C$ до частоти тактового сигналу приймача $R \times C$ (1:1, 1:16 та 1:64);

- розряди D_3, D_2 – визначають кількість бітів даних, якщо довжина символу менш ніж вісім бітів, невикористаними будуть старші біти, які під час читання буферу приймача дорівнюватимуть нулю;

- розряди D_4 – дозвіл або заборона встановлення розряду паритету;

- D_5 – за умови $D_4 = 1$ визначає режим контролю на парність/непарність;

– розряди $D6$, $D7$ – визначають кількість стоп-бітів для передачі.

Передавач автоматично додає до розрядів даних задану кількість старт-бітів, біти паритету (якщо $D4 = 1$) та задану кількість стоп бітів під час формування кадру даних відповідно до рис. 3.26.

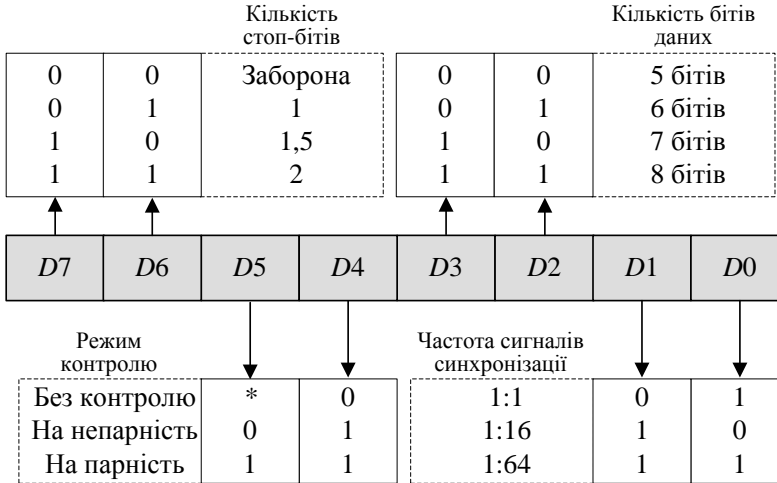


Рис. 3.27. Формат управляючого слова режиму роботи для асинхронного режиму

Формат УСРР для синхронного режиму обміну даними зображено на рис. 3.28.

Призначення розрядів управляючого слова режиму роботи для синхронного режиму обміну даними наступне:

- розряди $D0$, $D1$ – ($D0, D1 = 0$) завжди дорівнюють нулю для забезпечення синхронного режиму;
- розряд $D6$ – визначає режим зовнішньої ($D6 = 1$) або внутрішньої ($D6 = 0$) синхронізації;
- розряд $D7$ – визначає використання одного ($D7 = 1$) або двох ($D7 = 0$) символів синхронізації;
- розряди $D2$, $D3$, $D4$, $D5$ – мають теж саме значення, що й для асинхронного режиму.

В асинхронному режимі обміну даними у ПЗА завантажуються тільки регістр УСРР і регістр УСК. В синхронному режимі окрім регістрів УСРР та УСК завантажуються один або два символи синхронізації.

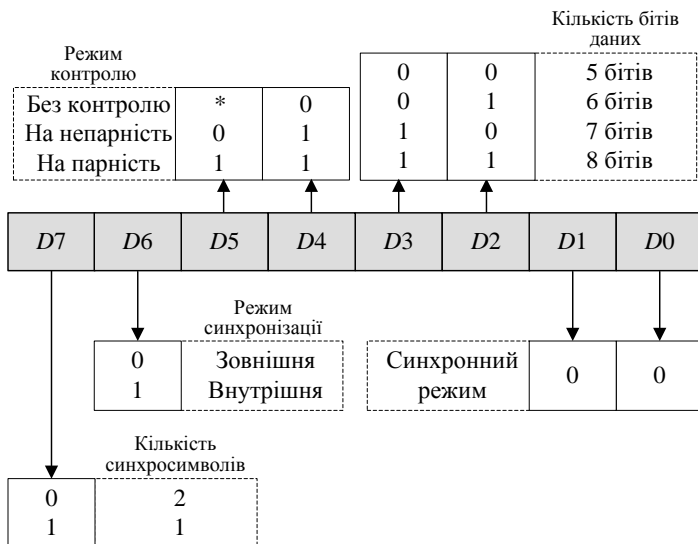


Рис. 3.28. Формат управляючого слова режиму роботи для синхронного режиму

Формат управляючого слова команди

Управляюче слово команди подається після програмування режиму роботи та запису в регістри приймача синхросимволів при синхронному режимі роботи. Формат управляючого слова команди зображений на рис.3.29.

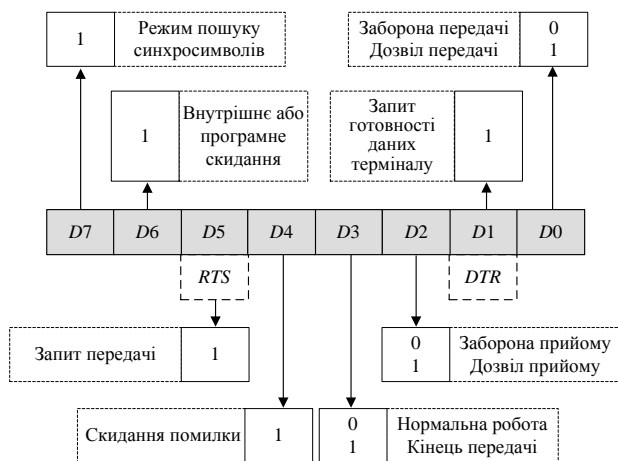


Рис. 3.29. Формат управляючого слова команди

- Призначення розрядів управляючого слова команди наступне:
- розряд $D0$ – дозвіл передачі даних;
 - розряд $D1 = DTR$ – завдання значення $\overline{DTR} = 0$ сигналу за-
питу готовності даних терміналу, якщо $D1 = 1$;
 - розряд $D2$ – дозвіл прийому;
 - розряд $D3$ – завдання нормального режиму асинхронної пе-
редачі ($D3 = 0$) або паузи ($D3 = 1$) під час відсутності даних для пе-
редачі, вихід передавача $\overline{T \times C}$ встановлюється під час цього у стан
0;
 - розряд $D4$ – скидання ознак помилок в управляючому слові
стану;
 - розряд $D5$ – стан входу RTS , завдання значення $\overline{RTS} = 0$ сиг-
налу запиту передачі даних, якщо $D5 = 1$;
 - розряд $D6$ – внутрішнє скидання ПЗА для переведу його в
режим очікування управляючого слова режиму роботи;
 - розряди $D7$ – дозвіл пошуку синхросимволів.

Формат управляючого слова статусу.

Формат управляючого слова статусу приводиться на рис. 3.30.

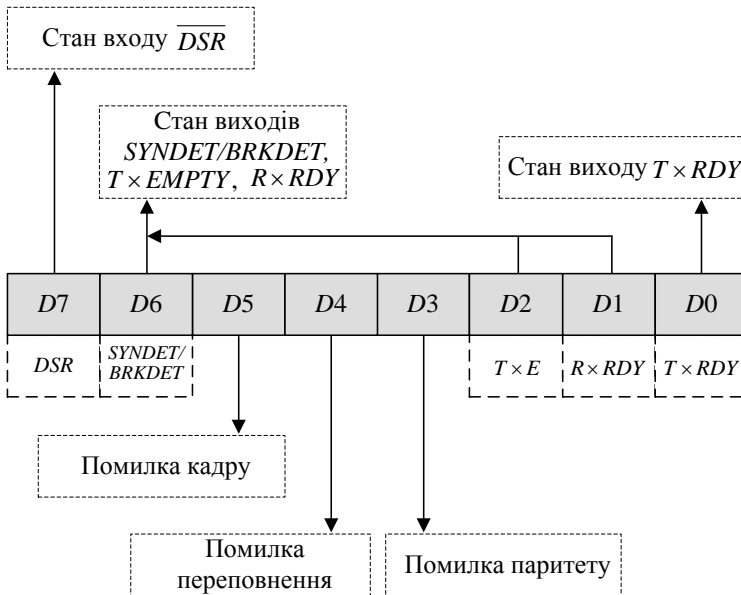


Рис. 3.30. Формат управляючого слова статусу

Призначення розрядів управляючого слова команди наступне:

- розряд $D0 = T \times RDY$ – на відмінність від виходу $T \times RDY$ значення даного сигналу не переводиться в ноль сигналом $\overline{CTS} = 1$ і значенням розряду $D0$ УСПР;
- розряд $D1$ – стан виходу $T \times RDY$;
- розряд $D2$ – стан виходу $T \times EMPTY$;
- розряд $D3$ – помилка паритету;
- розряд $D4$ – помилка переповнення;
- розряд $D5$ – помилка кадру (застосовується тільки для асинхронного режиму);
- розряд $D6$ – стан виходу $SYNDET/BRKDET$;
- розряди $D7 = DTR$ – стан виходу \overline{DTR} (дані для терміналу підготовлені).

Широкі можливості використання адаптеру, які надаються форматом слова УСПР, обумовлюють використання ПЗА в мікропроцесорних системах різної складності і з різноманітними характеристиками інтерфейсу. Під час асинхронних передач є можливість виключити символи синхронізації, що дозволяє прикладній програмі малої мікропроцесорної системи використовувати переваги більш складної організації інтерфейсу і в той же час не бути обтяжливою непотрібним вантажем надлишкових команд.

Структурна схема підключення програмованого зв'язуючого адаптера KP580BB51 до мікроконтролера МК48 приведена на рис. 3.31. Представлена мікропроцесорна система складається з зовнішньої пам'яті даних, програмованого зв'язуючого адаптера, регістру адреси, дешифратора для вибору однієї з сторінок ЗПД. Оскільки адресний простір має об'єм більш ніж 256 Кб реалізується сторінкова організація ЗПД. На структурній схемі рис. 3.31 показаний спосіб підключення до МК48 n сторінок зовнішньої пам'яті даних з використанням k виходів порту $P1$ (де $k = \lceil \log_2 n \rceil$) і дешифратора DC .

Адреси портів ПЗА входять у загальний адресний простір зовнішньої пам'яті даних. Для уникнення перетину адрес загального адресного простору ЗПД та ПЗА можна застосовувати селектор адреси CA (аналогічно, як на схемі рис. 3.21).

Доступ до портів під час запису та читання здійснюється за застосування команд $MOVX A, @Rr$; $MOVX @Rr, A$ (де, $r = 1, 0$).

Під час звернення до адрес у межах однієї сторінки ЗПД використовується одна команда `MOVX`. Переключення між сторінками потребує застосування додаткових команд.

В більшості випадків при підключенні до МК48 інтегральних схем адаптерів можна обійтись без додаткового устаткування (регістрів, дешифраторів і таке інше). Способи безпосереднього поєднання виходів МК48 з адаптерами КР580ВВ55 та КР580ВВ51 зображені на рис. 3.32.

На схемі, зображеній на рис. 3.32, *a* мається на увазі, що ЗПД складається тільки з регістрів ІС КР580ВВ55. Адреси регістрів задаються двома розрядами порту *P1*, тобто вміст регістрів *R0* та *R1* вибраного банку регістрів впливає на виконання команди `MOVX`.

Спосіб підключення КР580ВВ51 потребує декількох команд вводу-виводу для звернення до регістрів, однак система в цьому випадку містить зовнішню пам'ять даних та інше обладнання, пов'язане з шиною *BUS*.

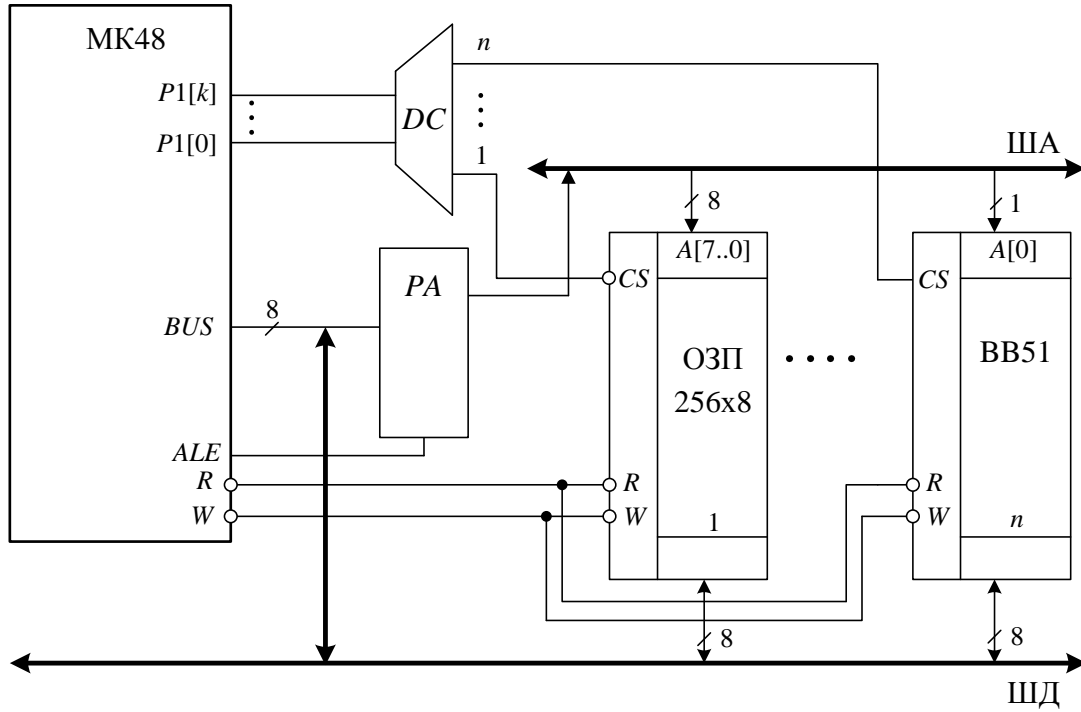


Рис. 3.31. Структурна схема підключення програмованого зв'язуючого адаптера КР580BB51 до мікроконтролера МК48

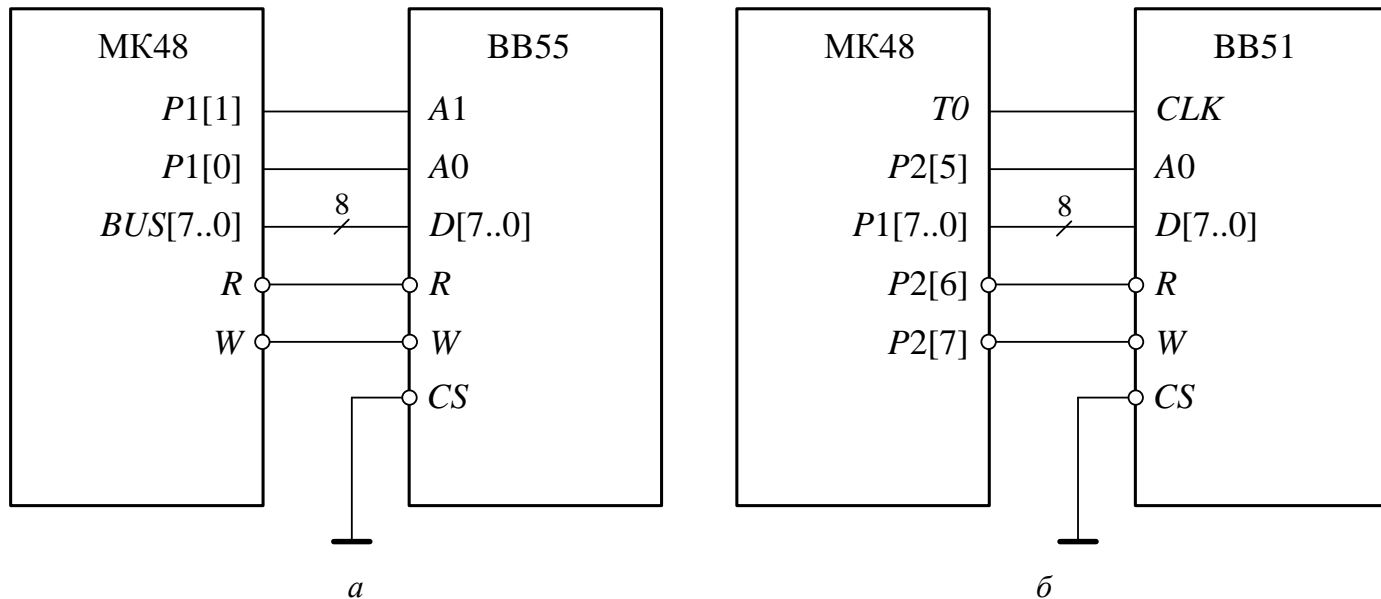


Рис. 3.32. Схеми підключення адаптерів до МК48: *a* – підключення інтегральної схеми МК580BB55; *б* – підключення інтегральної схеми МК580BB51

3.4. Система команд КР1816ВЕ48

3.4.1. Формат команд

Всі команди МК мають формат 1 чи 2 байта і виконуються за один чи два машинні цикли. Кожний цикл виконується за 5 тактів. Частота синхронізації тактів складає $F/3$, а циклів – $F/15$. Наприклад, при заданій частоті $F=6$ МГц тривалість тактів і циклів складає 0,5 і 2,5 мкс відповідно. За два машинні цикли виконуються всі команди з безпосереднім операндом, команди введення-виведення, команди передачі управління і роботи з підпрограмами, а також команди пересилок $MOVX$, $MOVP$, $MOVPR$. Решта команд виконуються за один машинний цикл. В МК передбачена можливість суміщення виконання однієї команди і вибірки наступної, що може зменшити час виконання команди. Мікроконтролер оперує з командами чотирьох типів (рис. 3.33).

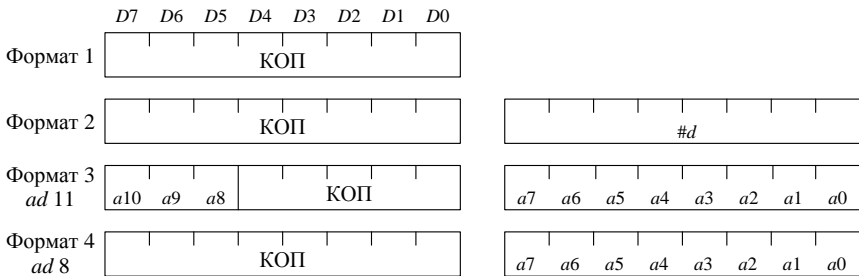


Рис. 3.33. Типи команд МК48

3.4.2. Система команд

У МК48 використовуються чотири способи адресації:

- пряма,
- безпосередня,
- непряма,
- неявна.

До переваг системи команд МК48 можна віднести: ефективний ввід/вивід, включаючи маскування і можливість управління окремими бітами портів; реалізацію розгалуження за значенням окремих бітів; обробку як двійкових, так і десяткових **двійково-кодуючих** чисел.

Під час виконання команд використовуються ознаки, що формуються в регістрі слова стану програми *RSW*, і ознаки користувача. Команди, в результаті виконання яких модифікуються ознаки, наведені в табл. 3.4. Функціональне призначення та способи формування ознак описані в розділі 3.2.2.

Таблиця 3.4. Формування ознак результату

Мнемоніка	Ознаки	Мнемоніка	Ознаки
ADD	<i>C AC</i>	JTF	<i>TF = 0</i>
CLR <i>C</i>	<i>C = 0</i>	MOV <i>PSW, A</i>	<i>C AC F0 BS</i>
CPL <i>C</i>	<i>C</i>	RETR	<i>C AC F0 BS</i>
CLR <i>F0</i>	<i>F0 = 0</i>	RLC <i>A</i>	<i>C</i>
CLR <i>F1</i>	<i>F1 = 0</i>	RRC <i>A</i>	<i>C</i>
CPL <i>F0</i>	<i>F0</i>	SEL <i>MB0; SEL MB1</i>	<i>DBF</i>
CPL <i>F1</i>	<i>F1</i>	SEL <i>RB0; SEL RB1</i>	<i>BS</i>
DA <i>A</i>	<i>C AC</i>		

Для опису команд використовуються мнемокоди мови асемблера МК48. Під час запису символічного коду команд застосовуються наступні позначення:

- A* — акумулятор;
- T*
- r* — номер регістру,
- Rr* — регістр з номером *r*;
- b* — номер біту,
- p* — номер порту вводу/виводу,
- Pp* — порт з номером *p*
- a* — адреса,
- d* — безпосередній операнд.
- #d* — безпосередній операнд (восьмирозрядне двійкове число);
- @Rr* — операнд, що адресується непрямо через *Rr*.
- @A*
- #Rr*

В стовбці «Коментарі» наведений опис операцій, що виконуються під час виконання команди. Для запису операцій використовуються мова мікрооперацій із застосуванням, символічних імен і скорочень. Номери розрядів регістрів подаються у квадратних дужках []. Операнд за непрямою адресацією подається у дужках (). Наприклад, запис *A := (Rr)* означає, що в акумуляторі *A* фіксується число,

що зчитане з внутрішньої пам'яті даних за адресою, записаною в регістрі Rr . Для запису операндів часто використовуються складені слова, що записані у вигляді двох слів, розділених крапкою. Наприклад, запис $PC[11..8].A$ подає дванадцятирозрядне двійкове слово, вісім молодших розрядів якого є вмістом акумулятора A , а чотири старші розряди – вмістом бітів (11..8) лічильника команд PC .

Всю множину команд асемблеру МК48 можна розбити на чотири основні групи:

- команди пересилки даних,
- команди основної групи:
 - виконання арифметичних операцій;
 - виконання логічних операцій;
- команди передачі управління;
- команди управління режимами роботи.

Система команд наведена в табл. 3.5. Команди згруповані за функціональними ознаками.

Команди пересилки даних

На рис. 3.34 представлений граф, що ілюструє можливі операції пересилки даних в МК48 (табл. 3.5). Операнди розрізняються за місцем розташування і за способом адресації. В операціях пересилки приймають участь такі операнди: акумулятор, РЗП, RSW, таймер, порти вводе/виводу, безпосередній операнд, ЗПД, РПД, ПП.

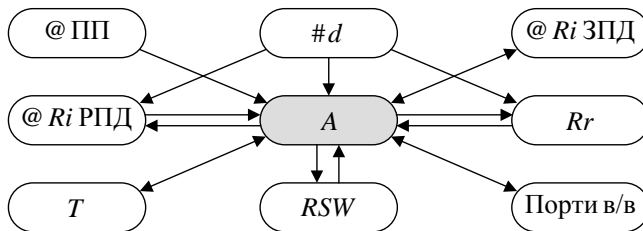


Рис. 3.34. Граф обміну даними у МК48

Всі операції пересилки даних в МК48 виконуються з застосуванням акумулятора A .

Під час пересилки даних між A та регістрами загального призначення РЗП банків регістрів, використовується пряма адресація, коли адреса операнда вміщується в тілі команди. Номер регістра, вміст якого пересилається в акумулятор вказується в трьох молодших бітах коду операції (рис. 3.33).

Таблиця 3.5. Система команд мікроконтролера KM1816BE48

Мнемоніка	Код команди	Коментарі
1	2	3
<i>Основна група команд та команди пересилки даних</i>		
<i>Команди звернення до акумулятора</i>		
CLR A	00100111	Встановлення вмісту акумулятора в нуль $A := 0$
CPL A	00110011	Інвертування вмісту A; $A := NOT A$
INC A	00010111	Інкремент вмісту A; $A := A + 1$
DEC A	00000111	Декремент вмісту A; $A := A - 1$
RR A	01110111	Циклічний зсув вмісту A вправо; $A[7] := A[0]$; $A[i] := A[i + 1]$; $i = \overline{6, 0}$
RL A	11100111	Циклічний зсув вмісту A вліво; $A[0] := A[7]$; $A[i] := A[i + 1] := A[i]$; $i = \overline{6, 0}$
RRC A	01100111	Циклічний зсув вмісту A з бітом переносу вправо; $A[7] := C$; $C := A[0]$; $A[i] := A[i + 1]$; $i = \overline{6, 0}$
RLC A	11110111	Циклічний зсув вмісту A з бітом переносу вліво; $A[0] := C$; $C := A[1]$; $A[i + 1] := A[i]$; $i = \overline{6, 0}$
SWAP A	01000111	Обмін тетрадами A; $A[7..0] \leftrightarrow A[3..0]$
DA A	01010111	Десяткова корекція вмісту A

Продовження табл. 3.5

1	2	3
MOV A, Rr ;r=(7-0)	11111rrr	Пересилка вмісту регістру в A; $A := Rr$
MOV Rr, A ;r=(7-0)	10101rrr	Пересилка вмісту A в регістр; $Rr := A$
XCH A, Rr ;r=(7-0)	00101rrr	Обмін вмісту A і регістру; $A \leftrightarrow Rr$
ANL A, Rr ;r=(7-0)	01011rrr	Логічне І вмісту A і регістру; $A := A \text{ AND } Rr$
ORL A, Rr ;r=(7-0)	01001rrr	Логічне АБО вмісту A і регістру; $A := A \text{ OR } Rr$
XRL A, Rr ;r=(7-0)	11011rrr	Виключне АБО вмісту A і регістру; $A := A \text{ XOR } Rr$
ADD A, Rr ;r=(7-0)	01101rrr	Сума вмісту A і регістру; $A := A + Rr$
ADDC A, Rr ;r=(7-0)	01111rrr	Сума вмісту A, регістру і переносу C; $A := A + Rr + C$
DEC Rr ;r=(7-0)	11001rrr	Декремент вмісту регістру; $Rr := Rr - 1$
INC Rr ;r=(7-0)	00011rrr	Інкремент вмісту регістру; $Rr := Rr + 1$
<i>Команди звернення до внутрішньої пам'яті даних</i>		
MOV A, @Rr ;r=0,1	1111000r	Пересилка із внутрішньої пам'яті даних в A; $A := (Rr)$
MOV @Rr, A ;r=0,1	1010000r	Пересилка вмісту A до внутрішньої пам'яті даних; $(Rr) := A$
XCH A, @Rr ;r=0,1	0010000r	Обмін вмістом A і комірки внутрішньої пам'яті даних; $A \leftrightarrow Rr$
XCHD A, @Rr ;r=0,1	0011000r	Обмін молодшими тетрадами A і комірки внутрішньої пам'яті даних; $A[3..0] \leftrightarrow (Rr[3..0])$
ANL A, @Rr ;r=0,1	0101000r	Логічне І вмісту A і комірки внутрішньої пам'яті даних; $A := A \text{ AND } (Rr)$

Продовження табл. 3.5

1	2	3
ORL A, @Rr ; r=0,1	0100000r	Логічне АБО вмісту A і комірки резидентної пам'яті даних; $A := A OR (Rr)$
XRL A, @Rr ; r=0,1	1101000r	Виключення АБО вмісту A і комірки резидентної пам'яті даних; $A := A XOR (Rr)$
ADD A, @Rr ; r=0,1	0110000r	Сума вмісту A і комірки резидентної пам'яті даних; $A := A + (Rr)$
ADDC A, @Rr ; r=0,1	0111000r	Сума вмісту A, комірки резидентної пам'яті даних і переносу C; $A := A + (Rr) + C$
INC @Rr ; r=0,1	0001000r	Інкремент комірки резидентної пам'яті даних; $(Rr) := (Rr) + 1$
<i>Команди роботи з зовнішньою пам'яттю даних</i>		
MOVX A, @Rr ; r=0,1	1000000r	Пересилка із ЗПД в A; $A := (Rr)$
MOVX @Rr, A ; r=0,1	1001000r	Пересилка вмісту A до ЗПД; $(Rr) := A$
<i>Команди звернення до пам'яті програми</i>		
MOV Rr, #d ; r=(7-0)	10111rrr dddddddd	Пересилка безпосереднього операнда до регістру; $(Rr) := d$
MOV A, #d	00100011 dddddddd	Пересилка безпосередньої адреси до A $A := d$
MOV @Rr, #d ; r=0,1	1011000r dddddddd	Пересилка безпосереднього операнда до внутрішньої пам'яті даних $(Rr) := d$

Продовження табл. 3.5

1	2	3
ANL A, #d	01010011 dddddddd	Логічне І вмісту A з безпосереднім операндом; $A := A \text{ AND } d$
ORL A, #d	01000011 dddddddd	Логічне АБО вмісту A з безпосереднім операндом; $A := A \text{ OR } d$
XRL A, #d	11010011 dddddddd	Виключне АБО вмісту A з безпосереднім операндом; $A := A \text{ XOR } d$
ADD A, #d	00000011 dddddddd	Сума вмісту A та безпосереднього операнду; $A := A + d$
ADDC A, #d	00010011 dddddddd	Сума вмісту A, безпосереднього операнду та переносу C; $A := A + d + C$
MOVP A, @A	10100011	Пересилка даних із поточної сторінки пам'яті програм до A; $A := (PC[11..8].A)$
MOV P3 A, @A	11100011	Пересилка даних із сторінки 3 пам'яті програм до A; $A := (0011 .A)$
<u>Команди звернення до регістру PSW</u>		
MOV PSW, A	11010111	Пересилка вмісту A до регістру PSW; $PSW := A$
MOV A, PSW	11000111	Пересилка вмісту регістру PSW до A; $A := PSW$
MOV A, T	01000010	Пересилка вмісту TCNT в A; $A := TCNT$
MOV T, A	01100010	Пересилка вмісту A в TCNT; $TCNT := A$
<u>Команди встановлення ознак</u>		
CLR C	10010111	Встановлення в нуль ознаки C; $C := 0$

Продовження табл. 3.5

1	2	3
CPL C	10100111	Інвертування ознаки C; $C := NOT C$
CLR F0	10000101	Встановлення в нуль ознаки F0; $F0 := 0$
CLR F1	10100101	Встановлення в нуль ознаки F1; $F1 := 0$
CPL F0	10010101	Інвертування ознаки F0; $F0 := NOT F0$
CPL F1	10110101	Інвертування ознаки F1; $F1 := NOT F1$
<i>Команди звернення до портів P1 і P2</i>		
ANL Pp, #d ; p = 1, 2	100110pp dddddddd	Логічне І порту P1 (P2) з безпосереднім операндом; $Pp := Pp AND d$
ORL Pp, #d ; p = 1, 2	100010pp dddddddd	Логічне АБО порту P1 (P2) з безпосереднім операндом; $Pp := Pp OR d$
IN A, Pp	000010pp	Введення даних із порту P1 (P2) в A; $A := Pp$
OUTL Pp, A	001110pp	Виведення вмісту A в порт P1 (P2) $Pp := A$
<i>Команди звернення до портів P4, P5, P6, P7</i>		
ANLD Pp, A ; p = (7 - 4)	100111pp	Логічне І порту P4 (P5, P6, P7) з A; $Pp := Pp AND A[3..0]$
ORLD Pp, A ; p = (7 - 4)	100011pp	Логічне АБО порту P4 (P5, P6, P7) з A; $Pp := Pp OR A[3..0]$
MOVD A, Pp ; p = (7 - 4)	000011pp	Ввід із порту P4 (P5, P6, P7) в A; $A[7..4] := 0$; $A[3..0] := Pp$
MOVD Pp, A ; p = (7 - 4)	001111pp	Вивід молодшої тетради із A в порт P4 (P5, P6, P7); $Pp := A[3..0]$

Продовження табл. 3.5

1	2	3
<i>Команди звернення до порту BUS</i>		
ANL BUS, #d	10011000 dddddddd	Логічне І порту <i>BUS</i> з безпосереднім операндом; <i>BUS := BUS AND d</i>
ORL BUS, #d	10001000 dddddddd	Логічне АБО порту <i>BUS</i> з безпосереднім операндом; <i>BUS := BUS OR d</i>
INS A, BUS	00001000	Введення даних із порту <i>BUS</i> в <i>A</i> ; <i>A := BUS</i>
OUTL BUS, A	00000010	Виведення вмісту <i>A</i> в порт <i>BUS</i> ; <i>BUS := A</i>
<i>Команди передачі управління</i>		
JMP a	aaa00100 aaaaaaaa	Безумовний перехід <i>PC[10..0] := a[10..0]</i> ; <i>PC[11] := MB</i>
JMP @A	10110011	Безумовний перехід в межах поточної сторінці; <i>PC[7..0] := (A)</i>
JC a	11110110 aaaaaaaa	Перехід, якщо <i>C = 1</i> <i>PC[7-0] := a</i> інакше <i>PC := PC + 2</i>
JNC a	11100110 aaaaaaaa	Перехід, якщо <i>C = 0</i>
DJNZ Rr, a	11101rrr aaaaaaaa	Декремент вмісту регістру і перехід, якщо вміст регістру не дорівнює нулю
JZ a	11000110 aaaaaaaa	Перехід, якщо вміст <i>A</i> дорівнює нулю
JNZ a	10010110 aaaaaaaa	Перехід, якщо вміст <i>A</i> не дорівнює нулю

Продовження табл. 3.5

1	2	3
JF0 a	10110110 aaaaaaaa	Перехід, якщо $F0=1$
JF1 a	01110110 aaaaaaaa	Перехід, якщо $F1=1$
JT0 a	00110110 aaaaaaaa	Перехід, якщо $T0=1$
JNT0 a	00100110 aaaaaaaa	Перехід, якщо $T0=0$
JT1 a	01010110 aaaaaaaa	Перехід, якщо $T1=1$
JNT1 a	01000110 aaaaaaaa	Перехід, якщо $T1=0$
JTF a	00010110 aaaaaaaa	Перехід, якщо $TF=1$
JNI a	10000110 aaaaaaaa	Перехід, якщо $INT=0$
JBb a	bbb10010 aaaaaaaa	Перехід, якщо розряд Bb акумулятора встановлений в одиницю
CALL a	aaa10100 aaaaaaaa	Виклик підпрограми; $SP:=SP+1$; $(SP):=PSW[7..4]$; $PC[11]:=MB$; $PC[10..0]:=a[10..0]$
RET	10000011	Повернення із підпрограми; $SP:=SP-1$; $PC:=(SP[11..0])$

Продовження табл. 3.5

1	2	3
RETR	10010011	Повернення із підпрограми з встановленням стану; $SP := SP - 1$; $PC := SP[11..0]$; $PSW[7..4] := (SP[15..12])$
Команди управління режимами роботи		
ENTO CLK	01110101	Дозвіл видачі імпульсів синхрон. на <i>TO</i>
SEL MBO	11100101	Вибір нульового банку пам'яті програм; $MB := 0$
SEL MB1	11110101	Вибір першого банку пам'яті програм; $MB := 1$
SEL RBO	11000101	Вибір нульового банку регістрів пам'яті даних; $RB := 0$
SEL RB1	11010101	Вибір першого банку регістрів пам'яті даних; $RB := 1$
NOP	00000000	Немає операції
EN I	00000101	Дозвіл зовнішніх переривань
DIS I	00010101	Заборона зовнішніх переривань
EN TCNTI	00100101	Дозвіл переривань від таймера/лічильника
DIS TCNTI	00110101	Заборона переривань від таймера/лічильника
STRT T	01010101	Запускання таймера/лічильника в режимі таймера
STRT CNT	01000101	Запускання таймера/лічильника в режимі лічильника
STOP TCNT	01100101	Зупинка таймера/лічильника

Обмін даними між A та комірками РПД або ЗПД здійснюється з використанням непрямої адресації. При цьому, покажчики адреси розміщуються в регістрах $R0$ або $R1$ вибраного банку регістрів.

За неявної адресації у коді команди неявно (за замовчуванням) указується один з операндів. Найчастіше таким операндом є акумулятор. За безпосередньої адресації у тілі команди в якості другого байту вказується безпосередній операнд (константа), який пересилається за місцем призначення, визначеним першим операндом.

До пам'яті програм здійснюється доступ лише у напряму читання даних.

Всі команди, окрім $MOV\ PSW, A$, не впливають на встановлення ознак.

Більшість команд виконує пересилку восьмибітних даних. Деякі команди оперують з чотирибітними операндами (тетрадами) і застосовуються для звернення до чотирибітних портів вводу/виводу $P4, P5, P6, P7$.

В МК48 передача даних відбувається двох режимах: пересилки (завантаження) и обміну. Під час пересилки дані передаються від джерела до приймача, при цьому джерело даних не змінює свого вмісту. Обмін припускає одночасну передачу даних в обох напрямках при цьому змінюються значення обох операндів, що приймають участь в обміні.

Команди пересилки даних всередині МК48 виконуються за один машинний цикл, а обмін даними з ЗПД потребує двох машинних циклів.

Приклади команд:

$MOV\ A, Rr$; $(A) := Rr, r = (7 - 0)$; пряма адресація.
 $MOV\ A, \#d$; $A := d$; безпосередня адресація.
 $MOV\ A, \#05$; $A := 05$; безпосередня адресація.
 $MOV\ Rr, \#d$; $(Rr) := d, r = (7 - 0)$.
 $MOV\ A, PSW$; $(A) := PSW$.
 $MOV\ A, T$; $(A) := T$.
 $MOV\ A, @R0$; $(A) := ((Rr)), r = 0, 1$; непряма адресація.
 $XCH\ A, Rr$; $(A) \leftrightarrow (Rr), r = (7 - 0)$.
 $XCH\ A, \#Rr$; $(A) \leftrightarrow (Rr), r = (7 - 0)$.

Команди основної групи

До команд основної групи належать команди виконання арифметичних та логічних операцій. В МК48 виконуються наступні операції над восьмибітними цілими двійковими числами без знаку: двійкове додавання, двійкове додавання з урахуванням переносу, операції десяткової корекції, інкременту, декременту, зсуву, кон'юнкції, диз'юнкції тощо (табл. 3.5).

Дві логічні команди скидання CLR та інверсія CPL дозволяють виконувати операції з бітами.

Під час додавання застосовується неявна адресація джерела першого операнду і місця розташування результату, в якості яких використовується акумулятор. До вмісту акумулятора можна додавати вміст регістру R3П, константу, вміст комірки РПД. Під час підсумовування формується ознака переносу *C*, що фіксується у відповідному розряді регістру *RSW* (див. модель програміста на рис. 3.3). Команда підсумовування ADDC з урахуванням переносу дозволяє виконувати підсумовування багатобайтних чисел.

В МК48 відсутня безпосередньо операція віднімання, при цьому віднімання реалізується за наступної послідовності дій: отримання додаткового коду другого операнду, додавання його до вмісту *A*, де зберігається перший операнд, та подання результату в доповнювальному коді.

Складні арифметичні операції ділення, множення, піднесення до ступеню і таке інше виконуються за підпрограмами.

Команди виконання арифметичних операцій змінюють відповідні ознаки в регістрі слова стану програми *RSW* (табл. 4.3).

Приклади команд:

ADD *A, Rr* ; $(A) := (A) + (Rr)$, $r = (7 - 0)$.

ADD *A, #d* ; $(A) := (A) + d$.

ADD *A, @Rr* ; $(A) := (A) + ((Rr))$, $r = 0, 1$.

ADDC *A, Rr* ; $(A) := (A) + (Rr) + C$, $r = (7 - 0)$.

INC *A* ; $(A) := (A) + 1$.

RL *A* ; $(A_{i+1}) := (A_i)$, $(A_0) := (A_7)$, $i = (7 - 0)$.

RLC *A* ; $(A_{i+1}) := (A_i)$, $(A_0) := (C)$, $(C) := (A_7)$, $i = (7 - 0)$.

RR *A* ; $(A_i) := (A_{i+1})$, $(A_7) := (A_0)$, $i = (7 - 0)$.

RRC *A* ; $(A_i) := (A_{i+1})$, $(A_7) := (C)$, $(C) := (A_0)$, $i = (7 - 0)$.

ANL	A, Rr;	; (A) := (A) & (Rr), r = (7 - 0).
ORL	A, @Rr;	; (A) := (A) ∨ (Rr), r = 0, 1.
XRL	A, #d;	; (A) := (A) ⊕ d.
CLR	A;	; (A) := 0.
CPL	A;	; (A) := \overline{A} .
CLR	C;	; (C) := 0.
CPL	F1;	; (F1) := $\overline{(F1)}$.

Принцип виконання зсувів командами RL, RLC, RR, RRC ілюструє схема, що показана на рис. 3.35.

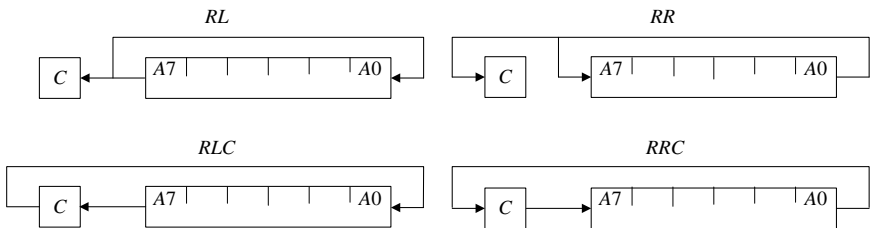


Рис. 3.35. Схема виконання операцій зсуву

Команди передачі управління

В групу команд передачі управління входять дві команди безумовного переходу, команди умовних переходів, команда виклику підпрограми і дві команди повернення з підпрограми.

Виконання команд передачі управління і роботи з підпрограмами має свої особливості і залежить від розподілу програм і даних за банками і сторінкам пам'яті.

Більшість команд передачі управління застосовують *пряму адресування*, при цьому в тілі команди вказується адреса переходу.

Команди *безумовного переходу* JMP і CALL вміщують одинадцятибітну адресу переходу і дозволяють здійснити перехід до будь-якої комірки ЗПП тільки в межах одного банку пам'яті програм (2048 адресів). Номер банку пам'яті програм визначається ознакою DBF, значення якого копіюється у старший біт лічильника команд (PC11) під час виконання команд JMP или CALL.

Перехід до іншого банку пам'яті програм можливий тільки за використання команд із групи команд управління режимами роботи (розділ 3.4.6) SEL MB1 та SEL MB1. До тих пір поки не буде обрано

інший блок пам'яті, всі переходи здійснюються в межах поточного банку пам'яті.

Інші команди є командами умовних переходів (окрім команд повернення), вміщують восьмибітну адресу переходу і дозволяють здійснювати передачу управління лише в межах поточної сторінки (256 байт). Така адресація називається *короткою прямою адресацією*, вона обумовлює певні обмеження на розміщення програм у пам'яті.

За застосування команд умовних переходів ознаки, що аналізуються, за виключенням C і $F0$ не фіксуються в спеціальних регістрах ознак, а подаються короткотривалими значеннями сигналів на виходах АЛП або відповідних входах МК48.

За застосування команд умовних переходів можна перевіряти не тільки внутрішні ознаки, але і деякі сигнали на зовнішніх входах МК48. Це дозволяє виконувати ефективно розгалуження в програмах без попереднього вводу і аналізу.

Непряма адресація реалізована командою $JMPR$, яка здійснює перехід за адресою, що зберігається у вказаній комірці ПП. Показником адреси є акумулятор. Перехід також здійснюється у межах однієї сторінки ПП.

Для організації програмних циклів використовується команда $DINZ$ (декремент регістру і перехід, якщо не нуль), яка дозволяє використовувати будь-який із восьми робочих регістрів в якості лічильника і здійснює перехід в межах поточної сторінки пам'яті програм.

Для роботи з підпрограмами використовуються команди $CALL$ та RET . Команда $CALL$, дозволяє звернутись до будь-якої комірки поточного банку ПП. Під час виклику підпрограми в стеку запам'ятовується адреса повернення і частина регістру RSW . Кількість вкладань обмежується глибиною стеку (16 байтів) і не повинна перевищувати восьми. Під час повернення з підпрограми в лічильнику команд поновлюється адреса повернення. Команда $RETR$ використовується для повернення з підпрограми обробки переривань, окрім адреси повернення поновлює вміст регістру RSW і встановлює дозвіл переривання від даного джерела.

Приклади команд:

$$\begin{aligned} JMP \quad a \quad ; (PC[10..8]) := A[10..8], \\ \quad \quad \quad ; (PC[7..0]) := A[7..0], (PS_{11}) := (MB) . \end{aligned}$$

DJNZ	Rr, a	; (Rr) := (Rr) - 1; якщо (Rr) ≠ 0, то ; (PC[7..0]) := a, в іншому випадку (PC) := (PC) + 2.
JC	a	; Якщо C = 1, то (PC ₀₋₇) := a, в іншому випадку ; (PC) := (PC) + 2.
JZ	a	; Якщо (A) = 0, то (PC ₀₋₇) := a, в іншому випадку ; (PC) := (PC) + 2.
JBb	a	; Перехід, якщо заданий біт A Bb дорівнює 1.
CALL	a	; Виклик підпрограми.
RETR		; Повернення з підпрограми з відновленням PSW.

Команди управління режимами роботи

В групу команд управління режимами роботи входять команди управління таймером/лічильником, перериваннями и ознаками переключення банків регістрів и банків пам'яті програм.

Вище були розглянуті команди обміну інформацією між таймером і акумулятором (MOV A, T и MOV T, A), за виконання яких може бути прочитано вміст таймера після зупинки підрахунку або безпосередньо під час підрахунку ("на льоту"), а також за необхідністю перезавантажено вміст таймера. Окрім цього в МК48 виконуються спеціальні команди управління режимом роботи таймера.

Таймер, залежно від застосованої команди, може бути використаний як лічильник тактів від внутрішнього джерела сигналів або як лічильник подій від зовнішнього джерела сигналів.

Система команд МК48 має в своєму розпорядженні засоби дозволу або заборони переривання від таймера. Спеціальною командою ENT0 на вивід T0 дозволяється передача імпульсів з частотою опорного синхросигнала, діленою на три. Видача цього сигналу може бути відключена тільки сигналом загального скидання. Синхросигнал на виході T0 використовується для загальної синхронізації зовнішніх пристроїв, узгоджених з МК48 за частотою роботи.

Приклади команд:

MOV	T, A	; Завантаження таймеру.
STRT	T	; Запуск таймеру.
STRT	CNT	; Запуск лічильника.
EN	TCNTI	; Дозвіл переривання від таймеру.
DIS	TCNTI	; Заборона переривань від таймеру.

3.5. Розробка програм обробки даних в МК48

3.5.1. Виконання команд передачі даних

Приклад 3.1: Завантажити в регістри $R5$ та $R6$ нульового банку регістрів число $1FFFh$:

```
SEL   RB0           ; вибір банку регістрів
MOV   R5, #1Fh     ; запис в R5 числа 1Fh
MOV   R6, #FFh     ; запис в R6 числа FFh
```

Приклад 3.2: Завантажити в комірки РПД з адресами 55 і 56 число $20A0h$:

```
MOV   R0, #55h     ; завантаження в R0 вказівника адреси
MOV   @R0, #20h    ; запис в комірку 55 числа 20h
INC   R0           ; +1 до вказівника адреси
MOV   @R0, #A0h    ; запис в комірку 56 числа A0h
```

Приклад 3.3: Завантажити в таймер число $(-5)_{\text{ДК}}$ без втрати вмісту акумулятора:

```
XCh   A, R2        ; обмін R2 і акумулятора
MOV   A, #FBh     ; завантаження в A числа (-5)ДК
MOV   T, A        ; пересилка в таймер
XCh   A, R2        ; обмін R2 та A
```

Приклад 3.4: Переслати вміст регістрів $R2$, $R3$, $R7$ першого банку регістрів в ЗПД, починаючи з комірки $C0h$.

```
SEL   RB1         ; вибір першого банку регістрів
ANL   P1, #0      ; встановлення в нуль P1
ORL   P1, #4h     ; вибір другої сторінки ЗПД
MOV   A, R2       ; A := R2
MOV   R0, #C0h    ; R0 := C0h адреса комірки
MOV   X @R0, A    ; пересилка R2 в комірку з адресою C0h
INC   R0          ; +1 до вказівника адреси
MOV   A, R3       ; A := R3
```

```

MOV X    @R0, A    ; пересилка R3 в комірку з адресою C1h
INC      R0        ; +1 до вказівника адреси
MOV      A, R7     ; A := R7
MOV X    @R0, A    ; пересилка R7 в комірку з адресою C2h

```

3.5.2. Виконання команд арифметичних та логічних операцій

Приклад 3.5: Виконати додавання вмісту регістру R5 нульового банку регістрів і вмісту комірки РПД з адресою 56:

```

SEL      RB0       ; вибір першого банку регістрів
MOV      R1, #56   ; завантаження вказівника адреси
MOV      A, R5     ; A := R5
MOV      @R1, A    ; пересилка R5 в комірку з адресою 56

```

Приклад 3.6: Виконати додавання багатобайтних чисел, розміщених в РПД. Регістри R1 та R0 першого банку регістрів вказують початкові адреси, розміщення в РПД молодших байтів вихідних аргументів.

```

; додавання Z = W + Y
; R0 – початкова адреса W
; R1 – початкова адреса Y
; R2 – довжина W та Y
      SEL      RB1       ; вибір першого банку регістрів
      CLR      C         ; скидання прапору переносу
LOOP: MOV      A, @R0    ; завантаження поточного байту
      ADDC    A, @R1    ; додавання (W+Y)
      DA      A         ; корекція
      MOV      @R0, A    ; розміщення чергового байту
                        ; результату
      INC     R0        ; +1 до покажчика адреси
      INC     R1        ; +1 до покажчика адреси
      DYNZ   R2, LOOP  ; декремент R2 та повернення на
                        ; початок циклу поки R2
                        ; не дорівнюватиме нулю.

```

Приклад 3.7: Знайти різницю двох двобайтних чисел без знаку.

Операнди розташовуються в комітках третьої сторінки ЗПД з наступними адресами: перший операнд – 20h, 22h та другий операнд – 44h, 45h. Результат розмістити на місце зменшувального.

```

; віднімання  $Z = W - Y$ 
; W – вміст комірок ЗПД з адресами 20h, 22h (<20h>, <22h>)
; Y – вміст комірок ЗПД з адресами 44h, 45h(<44h>, <45h>)
SEL      RB0      ; вибір нульового банку регістрів
ANL      P2, #0   ; завантаження  $P2 := 00000000$ 
ORL      P2, #8H  ; вибір третьої сторінки ЗПД
MOV      R0, #20H ; завантаження покажчика адреси
MOV      R1, #22H ;
MOVX     A, @R0   ;  $A := <20h>$ 
MOV      R5, A    ;  $R5 := W_{CT}$ 
MOVX     A, @R1   ;  $A := <22h>$ 
MOV      R4, A    ;  $R4 := W_{ML}$ 
MOV      R1, #44H ; завантаження вказівника адреси
MOV      R0, #45H ;
MOVX     A, @R0   ;  $A := <45h>$ 
CPL      A        ;  $A := \bar{A}$ 
CLR      C        ;  $C := 0$ 
ADDC     A, #1    ;  $A := A + 1 + C$  (доповнювальний код)
MOV      R7, A    ; пересилка вмісту A в R7 ( $Y_{ML}$ )
MOVX     A, @R1   ;  $A := <44h>$ 
CPL      A        ;  $A := \bar{A}$ 
ADDC     A, #0    ;  $A := A + C$ 
MOV      R6, A    ;  $R6 := Y_{CT}$ 
CLR      0        ;  $C := 0$ 
MOV      A, R4    ;  $A := R4 - R7$ 
ADDC     A, R7    ;  $R7 := Z_{ML}$ 
MOV      R7, A    ;
MOV      A, #R5   ;  $A := R5$ 
ADDC     A, R6    ;  $A := R5 - R6$ 
MOV      R6, A    ;  $R6 := Z_{CT}$ 

```

Приклад 3.8: Виконати маскування при введенні інформації із порту P1. Переслати в регістр R5 інформацію з виводів порту P1[1], P1[2], P1[5], P1[6], P1[7]

```
IN    A, P1           ; введення байту із порту 1
ANL  A, #11100110b  ; маскування
MOV  R5, A           ; A := R5
```

Приклад 3.9: Виконати логічний зсув вліво двобайтного слова, розміщеного в регістрах R5 і R6.

```
SEL  RBO           ; вибір нульового банку регістрів
CLR  C             ; C = 0
MOV  A, R6         ; A := R6
RLC  A             ; зсув молодшого байту
MOV  R6, A         ; пересилання результату
MOV  A, R5         ; A := R5
RLC  A             ; зсув старшого байту
MOV  R5, A         ; пересилка результату
```

Приклад 3.10: Видати вміст акумулятора в послідовному коді через нульовий вивід порту P1[0], залишаючи без змін решту біт порту P1. Передачу вести, розпочинаючи з молодшого біту.

```
                MOV    R1, #8           ; лічильник бітів
LOOP:           JBO    LABEL 1         ; перехід, якщо біт A[0]
                ; дорівнює одиниці
                ANL    P1, #11111110  ; скидання P1[0]
                JMP    LABEL 2         ;
LABEL1:         ORL    P1, #1          ; встановлення P1[0]
                JMP    LABEL 2         ; надлишкова команда для
                ; вирівнювання часу
                ; передачі 0 і 1
LABEL2:         RR     A               ; зсув A вправо (підготовка
                DYNZ   R1, LOOP        ; до передачі чергового біту)
```

3.5.3. Виконання команд передачі управління

Приклад 3.11: Передати управління за міткою *LABL1*, якщо перемикач банку регістрів (біт *PSW[4]*) дорівнює 1

```

MOV    A, PSW      ; пересилання PSW в акумулятор
JB4    LABL1       ; перехід, якщо A[4]=1
LABL1: MOV    A, R5 ; A := R5

```

Приклад 3.12: Обчислити значення функції $F = 2(R5 + R2)$. Якщо після виконання всіх дій ознака *C* дорівнюватиме одиниці, здійснити додавання одиниці до вмісту *R7*

```

SEL    RB0         ; вибір нульового банку регістрів
MOV    A, R2       ; A := R2
CLR    C           ; C = 0
ADD    A, R5       ; A := R2 + R5
RLC    A           ; зсув вліво
MOV    R5, A       ; R5 := A
JC     DD1         ; якщо C=1, то перехід
JMP    DD2         ;
DD1:   INC    R7   ; R7 := R7 + 1
DD2:   NOP

```

Приклад 3.13: Встановити вивід порту *P2[7]* в одиницю, за надходження на вхід *T0* послідовності з восьми нульових імпульсів.

```

MOV    P5, #8      ; завантаження в R5 числа
                          ; імпульсів
LABEL1: JT0    LABEL1 ; очікування сигналу 0 на вході T0
LABEL3: JT0    LABEL2 ; чекання сигналу 1 на вході T0
        JMP    LABEL3 ;
LABEL2: DYNZ   R5, LABEL1 ; повторювати доки не поступить
                          ; восьмий імпульс
        ORL   P2, #80h ; встановлення одиниці на вході 7
                          ; порту P2

```


3.5.4. Розробка підпрограм виконання складних арифметичних операцій

Приклад 3.15: Розробити програму множення цілих восьмирозрядних чисел $Z = X \times Y$, де $(X, Y) < 1$. Множення реалізувати першим способом. У вихідному стані операнди подані в прямому коді.

Множення першим способом відбувається молодшими розрядами множника зі зсувом суми часткових добутків в сторону молодших розрядів за нерухомим множенням.

Операційна схема множення першим способом та алгоритм зображені на рис. 3.36 та рис.3.37. Цифрова діаграма стану регістрів зображена на рис. 3.38.

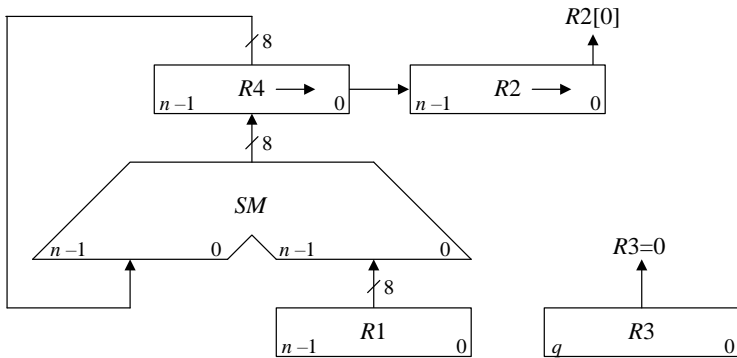


Рис. 3.36. Операційна схема множення чисел першим способом

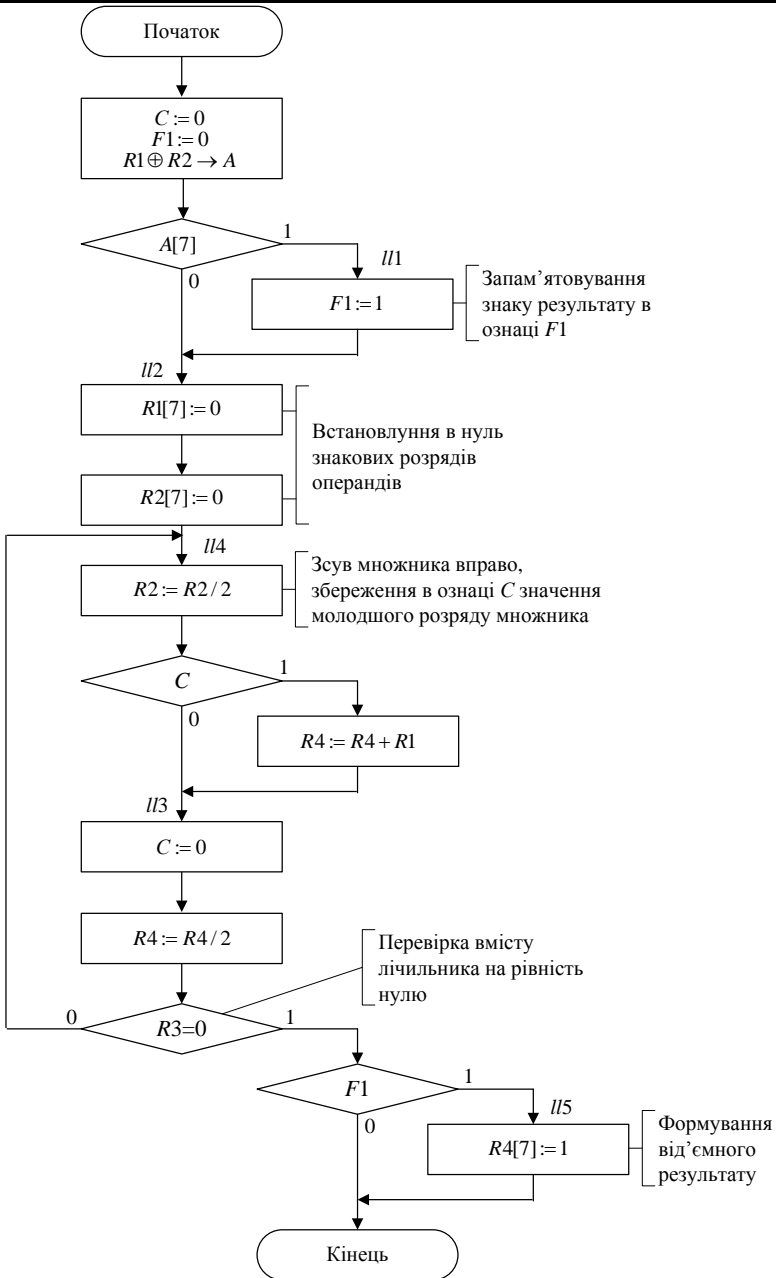


Рис. 3.37. Алгоритм виконання операції множення

	$\rightarrow R4$ [0 .. n]	$\rightarrow R2(X)$ [0 .. n]	$R1(Y)$ [0 .. n]	$R3$ (Лічильник)
	0 0000	0 1011	0 1111	101
1	0 0000 0 1111 0 1111		0 1111	
	0 0111	1 0101		100
2	0 0111 0 1111 1 0110		0 1111	
	0 1011	0 1010		011
3	0 0101	1 0101	0 1111	010
4	0 0101 0 1111 1 0100		0 1111	
	0 1010	0 1010		001
5	0 0101	0 0101	0 1111	000

Рис.3.38. Цифрова діаграма виконання операції множення

- ; У вихідному стані операнд знаходиться в регістрі $R1$ та $R2$.
- ; Після виконання операції множення старші розряди добутку збережені в регістрі $R4$, молодші – в регістрі $R2$.
- ; $R3$ – лічильник циклів.
- ; Визначення знаку результату і подання операндів в ПК

CLR C ; обнуління ознаки C

CLR F1 ; обнуління ознаки $F1$

XCH A, R1 ; обмін A та $R1$

XRL A, R2 ; ВИКЛЮЧНЕ АБО вмісту A та $R2$

JB7 l11 ; визначення знаку результату

JMP l12

L11: CPL F1 ; встановлення ознаки $F1$

L12: MOV A, R2 ; $A:=R2$

ANL A, #7Fh ; встановлення знакового розряду $R2$ в нуль

MOV R2, A ; $R2:=A$

MOV A, R1 ; $A:=R1$

```

    ANL    A, #7Fh    ; встановлення знакового розряду R1 в
                    ; нуль
    MOV    R1, A      ; R1:=A
; зсув множника вправо
L14:     MOV    A, R2    ; A:=R2
        CLRC
        RRC    A        ; зсув
        MOV    R2, A    ; R2:=A
        JNC    L13      ; аналіз цифри множника, перехід якщо
                    ; A[0]=0
; додавання множника до суми часткових добутків
        MOV    A, R4    ;
        ADD    A, R1    ; A := R4 + R1
        MOV    R4, A    ; R4 := A
; зсув суми часткових добутків
L13:     CLR    C        ; C = 0
        MOV    A, R4    ; A := R4
        RRC    A        ; зсув вправо, C := A[0]
        MOV    R4, A    ; R4:=A
; перевірка вмісту лічильника циклів, та
; повернення на початок циклу, якщо R3 ≠ 0
        DJNZ   R3, L14   ; аналіз лічильника циклів
; перевірка знаку результату, якщо F1 = 1 встановлення
; в одиницю старшого розряду регістру результату
        JF1    L15      ; аналіз знаку результату
        JMP    L16
L15:     MOV    A, R4    ; A:=R4
        ORL    A, #80h   ; R4[7]:=1
        MOV    R4, A    ; R4:=A
L16:     NOP
        END.

```

Приклад 3.16: Розробити програму перетворення цілого восьми-розрядного двійкового числа в трирозрядне двійково-десятькове число.

- ; Перед початком перетворення двійкове число знаходиться в R3.
- ; Після перетворення код старшої десяткової цифри знаходиться в

; молодших розрядах $R1$, а коди двох молодших десяткових цифр
; відповідно в старшій і молодшій тетradі регістру $R2$.

;

; початкові установки регістрів

INC R3

MOV R1, #0h

CLR A

JMP TST ; перехід на команду перевірки
; кінця циклу.

; цикл перетворення двійкового числа в двійково-десятькове

CYCLE: ADD A, #1h

DA A

JNC TST

INC R1

TST: DJNZ R3, CYCLE ; декремент $R3$ і перевірка кінця
; циклу.

MOV R2, A ; пересилка молодших розрядів
; результату в $R2$.

END.

Приклад 3.17: Розробити програму ділення правильних додатних двійкових дробів $Z=X/Y$, де $(X, Y) < 1$, $X < Y$. Ділення виконати способом зі зсувом дільника.

Операційна схема та алгоритм ділення представлені на рис. 3.39 та рис. 3.40, відповідно. Цифрова діаграма виконання операції ділення зображена на рис. 3.41.

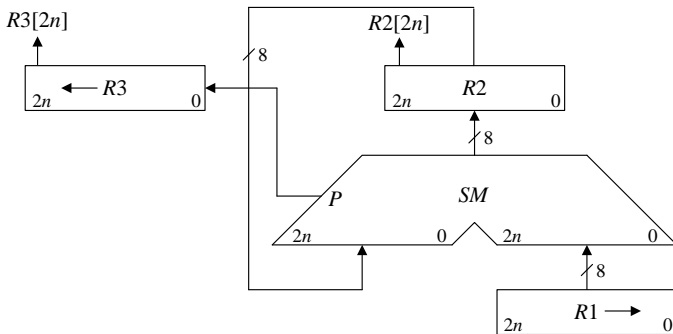


Рис.3.39. Операційна схема пристрою ділення із зсувом дільника

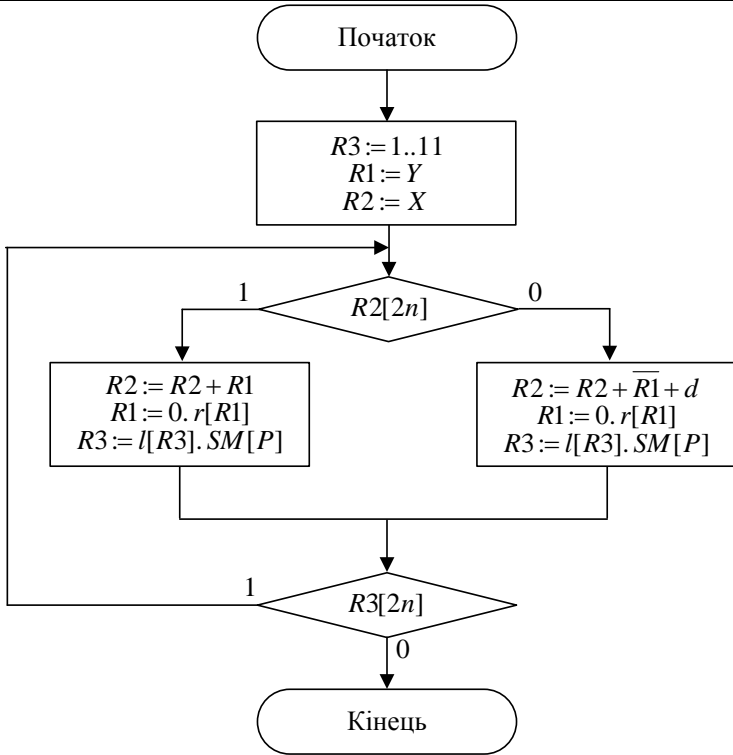


Рис. 3.40. Змістовний мікроалгоритм виконання операції ділення

№ такту	← R3			R2(X)			R1(Y) →			Логічна умова
	[2n	..	0]	[2n	..	0]	[2n	..	0]	
	1	11111111		0	1001000		0	11000000		
1				0	1001000					R1[2n]=0
				1	0100000					
	1	11101111		1	1101000		0	0110000		R2[2n]=1
2	1	11101111		0	0110000					R1[2n]=1
				0	0011000					
	1	11011111					0	0011000		R2[2n]=1
3	1	11011111		0	0011000		0	0011000		R1[2n]=0
				1	1101000					
	1	10111111		0	0000000		0	0001100		R2[2n]=1

4	1	10111111	0	0000000	0	0001100	$R1[2n]=0$
			1	1110100			
			1	1110100			
	1	01101111			0	0000110	$R2[2n]=1$
5	1	01101111	1	1110100	0	0000110	$R1[2n]=1$
			0	0000110			
			1	1111010			
	0	11001111			0	0000011	$R2[2n]=0$

Рис.3.41. Цифрова діаграма виконання операції ділення

;У вихідному стані Y завантажений в регістрі $R1$, X – в регістрі $R2$,
; В регістрі $R3$ завантажуються маркерні одиниці.
; Результат формується в $R3$, під час зсуву його вмісту вліво
; чергова цифра результату заноситься у розряд, що звільнився.
; Операція ділення завершується, коли під час зсуву $R3$ в ознаку C
; запишеться нуль.

```

SEL      RB0      ; вибір нульового банку регістрів
MOV      R3, #FFh; ; R3:=11111111
L15:    MOV      A, R2      ; A:=R2(X)
        JB7      L11      ; перевірка ознаки C (A[7])
        MOV      A, R1      ; пересилка R1 в A
        CLR      A          ; інвертування A
        ADD      A, #1      ; додавання одиниці до A
        ADD      A, R2      ; A:= (R2 + (R1 + 1))
        MOV      R2, A      ; пересилка діленого
        JMP      L13
L11:    MOV      A, R2      ; пересилка R2 в A
        ADD      A, R1      ; отримання суми діленого і дільника
L13:    MOV      R2, A      ; пересилка A в R2
        MOV      A, R1      ; пересилка R1 в A
        CPL      C          ; встановлення в нуль ознаки C
        RRC      A          ; зсув дільника
        MOV      R1, A      ; пересилка A в R1
        CPL      C          ; встановлення в нуль ознаки C
        MOV      A, R2      ; інвертування старшого
                                ; розряду дільника (R2)
        JB7      L14
        CLR      C

```

L14:	MOV	A, R3	; запис в молодший розряд
			; регістру R3
	RLC	A	
	MOV	R3, A	
	JB7	L15	
		NOF	
		END	

Приклад 3.21: Розробити для МК48 програму обчислення квадратного кореня $A = \sqrt{B}$, де $0 \leq B < 1$.

Найбільш простий алгоритм обчислення квадратного кореня з n -розрядної мантиси числа зводиться до підбору цифр результату розряд за розрядом, починаючи із старшого 2^{-1} розряду. При цьому обчислення i -ї цифри результату X відбувається таким чином. Після отримання чергової $(i-1)$ -ї цифри в i -й розряд A розміщується одиниця. Обчислюється різниця $(B - A_i^2) = R_i$. Якщо, $R_i \geq 0$ то $i A_i$ є число, де цифри всіх розрядів співпадають з цифрами результату A . Якщо, $R_i < 0$ то в i -му розряді необхідно поставити нуль і переходити до обчислення $(i+1)$ -го розряду. Оскільки в цьому випадку обчислення знову починається з підстановки пробної одиниці, то замість заміни одиниці на нуль в i -му розряді віднімається одиниця з $(i+1)$ -го.

Виконання обчислення з точністю до шостого розряду приведене на діаграмі (рис. 3.42, а). Для досягнення регулярності обчислень у разі отримання додатної різниці операцію віднімання доцільно замінити підсумуванням зворотного коду наступного результату з дописаними цифрами 11, при цьому отримуємо підсумовування в доповнювально коді, наприклад, діаграма (рис. 3.42, б).

Для виконання обчислень в МК48 застосовуються восьмирозрядні регістри, тому корінь з восьмирозрядної мантиси можливо обчислити тільки з точністю до чотирьох розрядів після коми.

Операційна схема пристрою для обчислення квадратного кореня зображена на рис. 3.43, цифрова діаграма та алгоритм обчислення на рис. 3.44, та 3.45 відповідно.

<pre> 0 00 10010001 - 00 01 ----- 1 00 01010001 00 10100010 зсув - 00 101 ----- 1 00 00000010 00 00000100 зсув - 00 1101 ----- 0 11 00110100 10 01101000 зсув + 00 11011 ----- 0 11 01000000 10 10000000 зсув + 00 110011 ----- 0 11 01001100 10 10011000 зсув + 00 1100011 ----- 0 11 01011110 A = √B = 0,110000 </pre>	<pre> 0 00 10010001 + 11 11 ----- 1 00 01010001 00 10100010 зсув + 11 011 ----- 1 00 00000010 00 00000100 зсув + 11 0011 ----- 0 11 00110110 10 01101100 зсув + 00 11011 ----- 0 11 01000100 10 10001000 зсув + 00 110011 ----- 0 11 01010100 10 10101000 зсув + 00 1100011 ----- 0 11 01011110 A = √B = 0,110000 </pre>
a	б

Рис. 3.42. Діаграма обчислення квадратного кореня: а – у прямому коді; б – у доповнювальному коді.

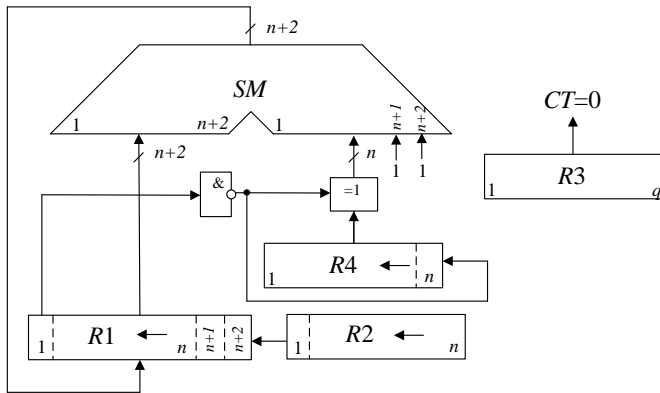


Рис. 3.43. Операційна схема обчислення квадратного кореня

№	← RA	← RB		← RC	CT	Мікрооперації
	[1 .. n]	[1 .. n]	$\frac{n+1}{n+2}$	[1 .. n]		
ПС	000000	000000	00	10010001	100	
1	000000	000000	10	01000100		$2(RGC := \lfloor RGC \rfloor.0,$ $RGB := \lfloor RGB \rfloor.RGC(0))$
		+111111	11			$RGB := RGB + RGA.11$
		000000	01			
2	00000, 1	000000	10	10001000		$RGC := \lfloor RGC \rfloor.0,$ $RGB := \lfloor RGB \rfloor.RGC(0),$ $RGA := \lfloor RGA \rfloor.RGB(0)$
		000001	01	00010000		$RGC := \lfloor RGC \rfloor.0,$ $RGB := \lfloor RGB \rfloor.RGC(0)$
		+111110	11			$RGB(0) = 0 \Rightarrow$ _____ $RGB := RGB + RGA.11$
		000000	00		011	$CT := CT - 1; CT \neq 0$
3	0000, 11	000000	00	00100000		$RGC := \lfloor RGC \rfloor.0,$ $RGB := \lfloor RGB \rfloor.RGC(0),$ $RGA := \lfloor RGA \rfloor.RGB(0)$
		000000	00	01000000		$RGC := \lfloor RGC \rfloor.0,$ $RGB := \lfloor RGB \rfloor.RGC(0)$
		+111100	11			$RGB(0) = 0 \Rightarrow$ _____ $RGB := RGB + RGA.11$
		111100	11		010	$CT := CT - 1; CT \neq 0$
4	000, 110	111001	10	10000000		$RGC := \lfloor RGC \rfloor.0,$ $RGB := \lfloor RGB \rfloor.RGC(0),$ $RGA := \lfloor RGA \rfloor.RGB(0)$
		110011	01	00000000		$RGC := \lfloor RGC \rfloor.0,$ $RGB := \lfloor RGB \rfloor.RGC(0)$
		+000110	11			$RGB(0) = 1 \Rightarrow$ _____ $RGB := RGB + RGA.11$
		111010	00		001	$CT := CT - 1; CT \neq 0$
5	00, 1100	110100	00	00000000		$RGC := \lfloor RGC \rfloor.0,$ $RGB := \lfloor RGB \rfloor.RGC(0),$ $RGA := \lfloor RGA \rfloor.RGB(0)$
		101000	00	00000000		$RGC := \lfloor RGC \rfloor.0,$ $RGB := \lfloor RGB \rfloor.RGC(0)$
		+001100	11			$RGB(0) = 1 \Rightarrow$ _____ $RGB := RGB + RGA.11$
		110100	11		000	$CT := CT - 1; CT = 0$

Рис.3.44. Цифрова діаграма обчислення квадратного кореня

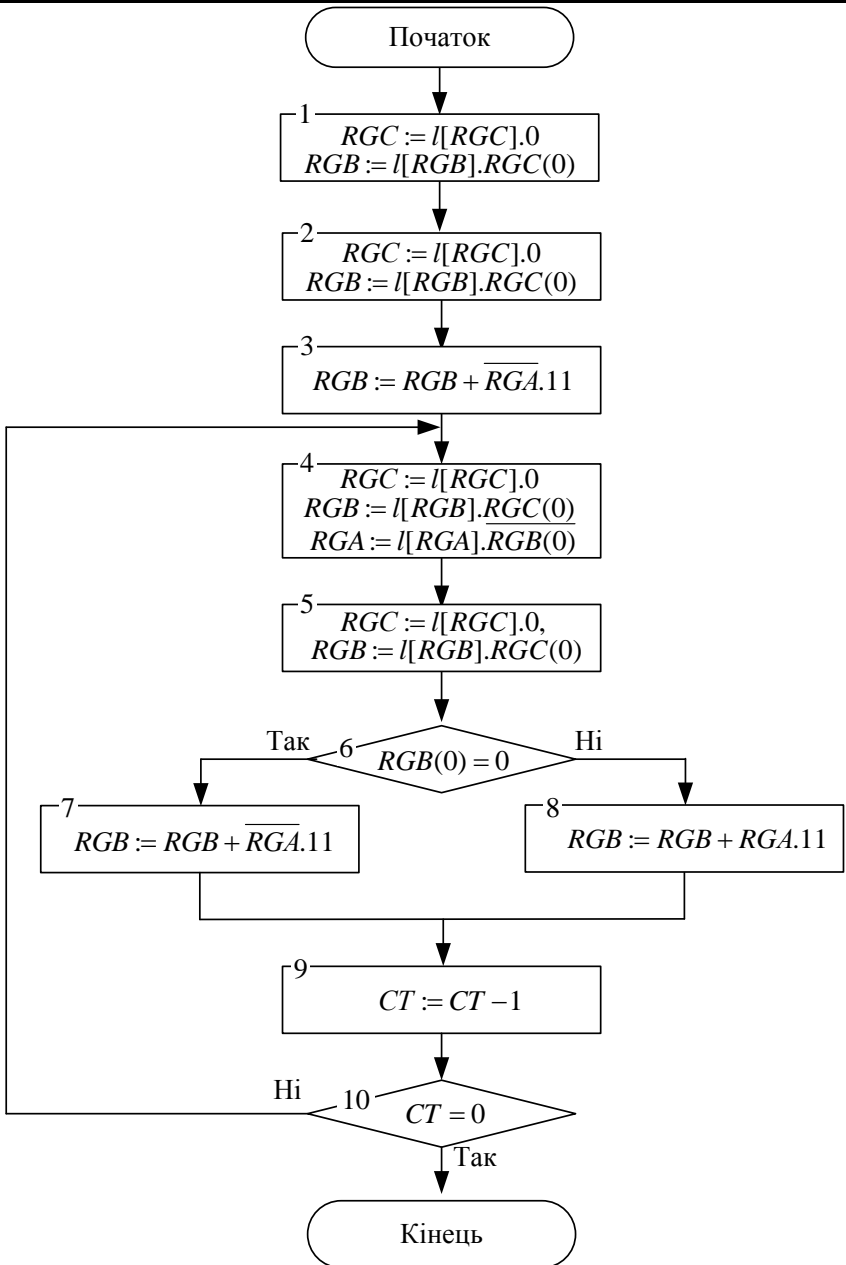


Рис. 3.45. Алгоритм обчислення квадратного кореня

```

; обчислення кореня з восьмирозрядної мантіси числа
; з точністю до чотирьох знаків після коми
    INS    A, BUS           ;
    MOV    R2, A           ; A – завантаження X
    MOV    R2, #10010001  ;
    B
    MOV    R1, #0h        ; B – для підсумовування
    MOV    R0, #0h        ; C – для
                                ; зберігання результату
; завдання кількості повторень циклу CT:=4
    MOV    R3, #4h
LL3:  MOV    A, R0
    MOV    R4, A           ; R4:=A(R4:=A.11)
    MOV    R5, #0h        ; B[ZN]:=0
; два зсуви вліво A, B
    CLR    C
    MOV    A, R2
    RLC    A
    MOV    R2, A
    MOV    A, R1
    RLC    A
    MOV    R1, A
    CLR    C
    MOV    A, R2
    RLC    A
    MOV    R2, A
    MOV    A, R1
    RLC    A
    MOV    R1, A
    JB7    LL1             ; аналіз B[ZN]
    MOV    A, R4
    CPL    A               ; R4:=!A
    MOV    R4, A
LL1:  MOV    A, R4
    RLC    A
    RLC    A
    ORL    A, #3h
    MOV    R4, A           ; R4:=A.11
    ADD    A, R1

```

```

MOV    R1, A
JB7    LL2
MOV    R5, #1h    ; !B[ZN]:=1
LL2:   CLR    C
MOV    A, R0
RLC    A
ADD    A, R5      ; A[N]:=!B[ZN]
MOV    R0, A
DJNZ   R3, LL3   ; перевірка циклу
MOV    A, R0
OUTL   BUS, A
END

```

3.5.5. Розробка програм управління

Приклад 3.14: Розробити програму реалізації алгоритму управління, що заданий логічною схемою алгоритм (ЛІСА).

Вихідні дані:

- Алгоритм управління:

$$H \ Y_5 X_1 \uparrow (Y_1 Y_2) X_2 \uparrow \downarrow (Y_3 Y_4) \downarrow K$$

- Тривалість управляючих сигналів:

$$T(y_1) = T(y_2) \geq 240 \text{ мкс}, \quad T(y_3) = T(y_4) \geq 30 \text{ мкс}, \quad T(y_5) = 450 \text{ мкс}$$

Алгоритм управління зображений на рис. 3.46.

Для вводу і виводу сигналів будемо використовувати порт $P1$, причому, розряди порту $P1[6]$ та $P1[7]$ в початковому стані налаштовані на ввід ($P1[6, 7] = 1$), а $P1[5..0]$ – на вивід інформації. Відповідність виходів порту і сигналів вказано в табл. 3.6.

Таблиця 3.6. Відповідність виходів порту і сигналів

Розряд порту	$P17$	$P16$	$P15$	$P14$	$P13$	$P12$	$P11$	$P10$
Сигнал	$X1$	Вхід	$Y1$	$Y2$	$Y3$	$Y4$	$Y5$	Вихід

Для формування потрібної тривалості сигналів $Y1$ і $Y2$ будемо використовувати $TCNT$ в режимі таймера. При $F=6 \text{ МГц}$ для відліку проміжку часу, не меншого 240 мкс, необхідна зміна змістовного $TCNT$ на 3. Для формування інтервалу часу, тривалістю 30 мкс (ке-

руючі сигнали $Y3$ та $Y4$), при вказаній частоті F потрібно виконати 12 командних циклів. Для формування затримки в 450 мкс ($Y5$), необхідно використовувати і таймер, і тривалість команди.

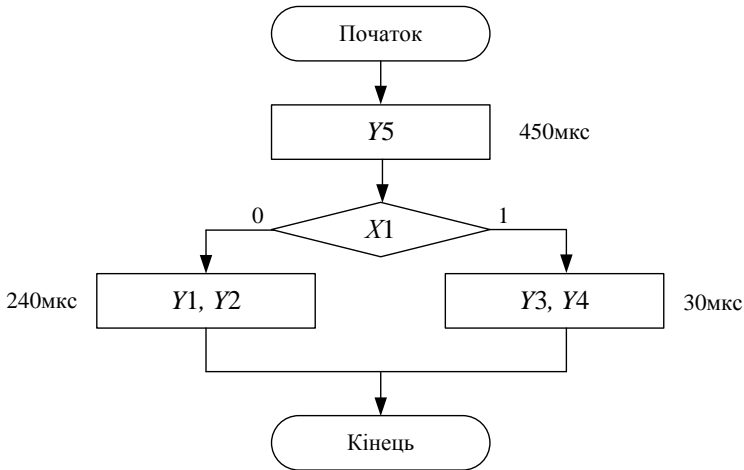


Рис. 3.46 Алгоритм управління

;Перед виконанням програми порт $P1$ знаходиться в стані 11111111.

```

MOV R5, #4H ; завантаження константи 4 в R5
MOV A, #FDH ; завантаження константи
; (-3)дк в таймер

MOV T, A
ORL P1, #18H ; встановлення сигналів
; Y3 = Y4 = 1
Loop: NOP ; відлік часового інтервалу
ANL P1, #C0h ; P1 := 11000000
MOV R5, #10 ; завантаження константи 10
; в R5
MOV A, #FBh ; завантаження константи
; (-5)дк в таймер

MOV T, A
ORL P1, #2 ; встановлення сигналу Y5 = 1
STRT T ; запуск таймера
Label2: JTF Label1 ; відлік часового інтервалу
JMP Label2 ; з використанням таймеру
; (400мкс)
Label1: DJNZ R5, label1 ; відлік часового інтервалу
  
```

			; з використанням R5(50мкс)
	ANL	P1, #C0h	; зняття управляючого сигналу
	MOV	A, #FDh	; завантаження константи
			; (- 3)дк в таймер
	MOV	T, A	
	MOV	R5, #4	; завантаження константи 4 в R5
	IN	A, P1	; ввід та перевірка
	JB7	Label3	
	ORL	P1, #30h	; встановлення сигналів
			; Y1 = Y2 = 1
	STRT	T	; запуск таймера
Label4:	JTF	Label5	; відлік часових інтервалів
	JMP	Label4	; з використанням таймеру
Label5:	ANL	P1, #C0h	; зняття керуючих сигналів
			; Y3 = Y4 = 1
Label3:	NOP		; відлік часового інтервалу
	DJNZ	R5, label3	; з використанням R5
	ANL	P1, #C0h	; зняття управляючих сигналів
	END.		

Приклад 3.18: Розробити структурну схему підключення до МК48 програмованого периферійного адаптера ВВ55. Розробити програму пересилки даних із порту *PB* в порти *PA* та *PC*. Адреси портів ППА належать загальному адресному простору зовнішньої пам'яті даних.

Вихідні дані:

- Адреси портів:

PA – *ACh*, *PB* – *ADh*, *PC* – *AEh*, Регістр УСРР – *AFh*.

Структурна схема підключення до МК48 однієї сторінки ЗПД та програмованого периферійного адаптера ВВ55 зображена на рис. 3.47. Для підключення ППА застосовується селектор адреси СА.

; Прийом байту із порту *PB* в порти *PA* та *PC*.

```
MOV      R0, #0AFh      ; адресу Регістру УСРР завантажуюмо у
                        ; покажчик адреси
```

```

MOV    A, #082h      ; ініціалізація BB55 (порти PA і PC
                    ; налаштовуються на вивід, а порт PB –
                    ; на ввід даних)

MOV    @R0, A
MOV    R0, #0ADh     ; підготовка адреси порту PB
MOVX   A, @R0        ; читання даних із порту PB
MOV    R0, #0ACh     ; підготовка адреси порту PA
MOV    @R0, A        ; запис даних в порт PA
MOV    @R1, #0AEh    ; підготовка адреси порту PC
MOV    @R1, A        ; запис даних в порт PC
END

```

Приклад 3.19: Розробити структурну схему підключення до МК48 ЗПД та програмованого зв'язувального адаптера BB51. Зовнішня пам'ять даних складається з чотирьох сторінок, для перемикавання між якими застосовуються два молодших виводи порту P1. Адреси регістрів УСРР та УСК ПЗА належать першій сторінці ЗПД (адреса РД -00h; адреса РС(УС) -01h)

Розроблена структурна схема МПС зображена на рис. 3.48.

; P1 = 11111111 встановлюється під час ініціалізації

```

ANL    P1, #0FDh     ; вибір номера сторінки
ORL    P1, #01h     ; "1" 11111101 для BB51
SEL    RB1          ; установка першого банку
                    ; регістрів

MOV    R0, #00h     ; адреса регістра даних ПЗА
                    ; завантажується в R0
MOV    R1, #01h     ; адреса Регістру УСРР (УСРР,
                    ; УСІ) ПЗА завантажується в R1
MOV    A, #07Eh     ; ініціалізація ПЗА
                    ; (асинхронний режим).
MOVX   @R1, A       ; передача байта в послідовний ;
                    ; канал з регістра R2
MOV    A, #31h     ; запис УСК (УСІ) в PG ПЗА
MOVX   @R1, A       ;
WW1:   MOVX   A, @R1 ; читання слова стану ПЗА
                    ; і перевірка готовності передавача

ANL    A, #01
JZ     WW1          ;

```

	MOV	A, R2	; запис даних в передавач ПЗА ; з R2
	MOVX	@R0, A	; прийом байта з послідовного ; каналу в R2
	MOV	A, #34h	; запис управляючого слова в ПЗА
	MOVX	@R1, A	;
WW2 :	MOV	X, a, @R1	; читання слова стану з ПЗА ; і перевірка готовності приймача
	ANL	a, #02h	
	JZ	WW2	
	MOVX	A, @R1	; читання стану слова в ПЗА ; і перехід, якщо знайдена помилка
	ANL	A, #38h	
	JNZ	ERROR	
	MOVX	A, @R0	; читання байта даних та передача ; в R2
	MOV	R2, A	;
ERROR :	NOP		; обробка помилки
	END		

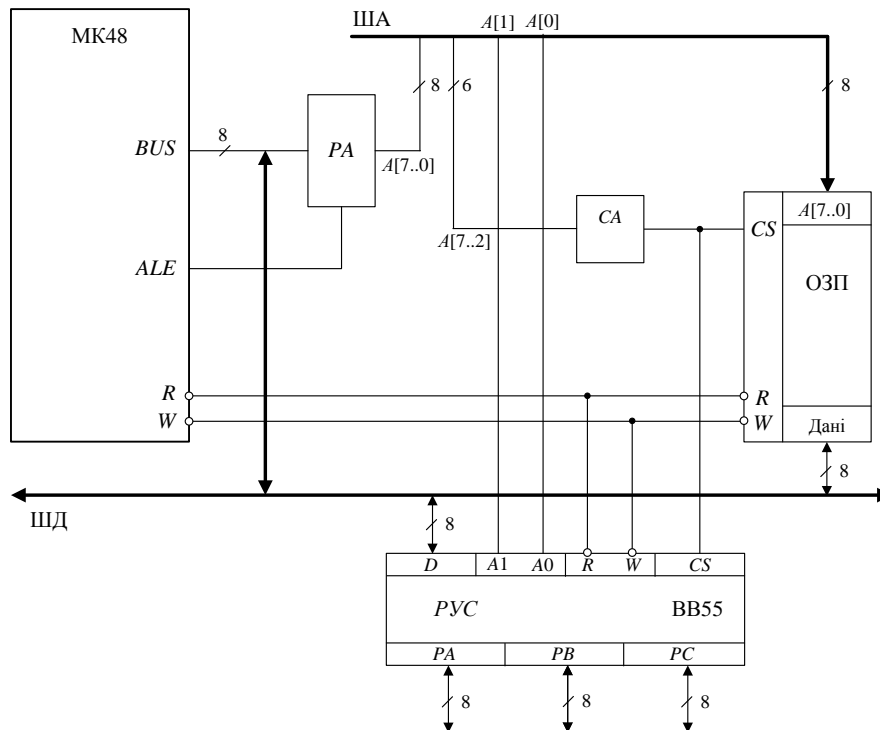


Рис. 3.47. Структурна схема підключення програмованого периферійного адаптера до МК48

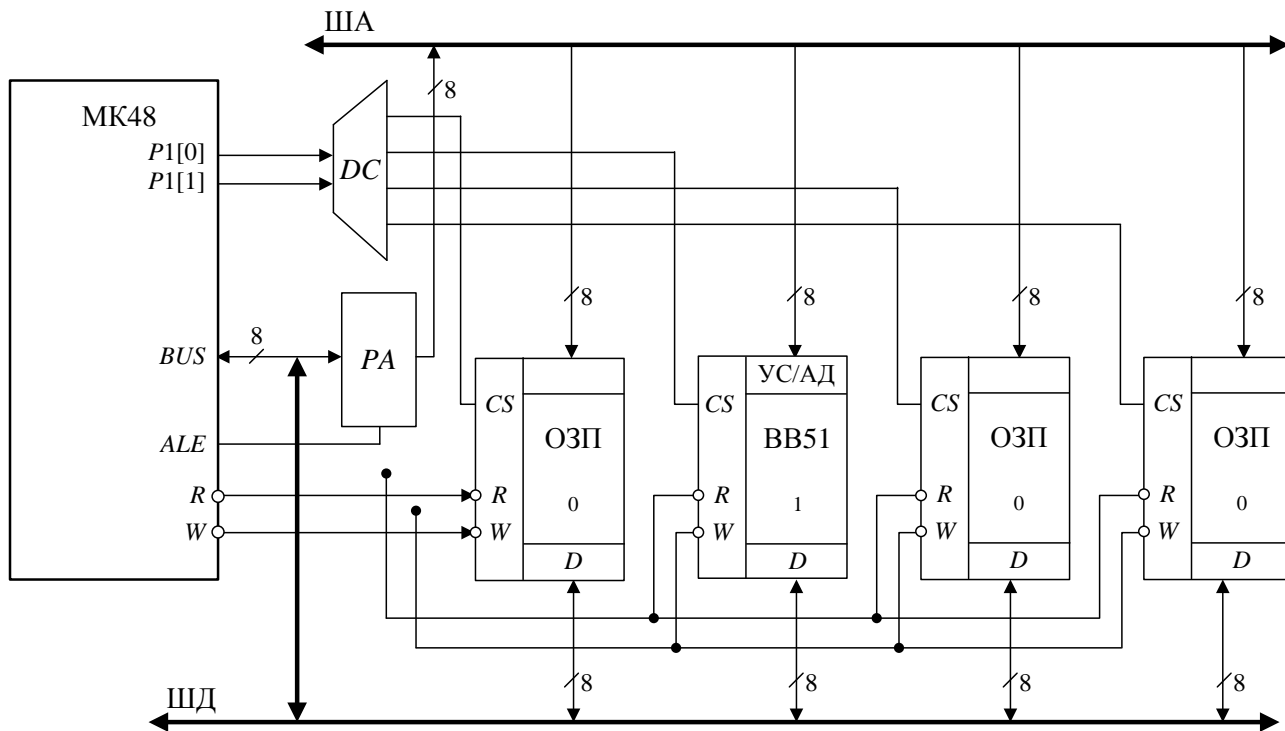


Рис. 3.48. Структурна схема підключення програмованого з'єднувального адаптера до МК48

Приклад 3.20: Розробити управляючу програму заборони переривання від таймеру та дозволу переривання після восьмого сигналу переповнення таймеру. Під час переходу до процедури обробки переривання зупинити таймер. Сигнали переповнення завантажити в *R5*.

```

STRT:  DIS    TCNTI    ; заборона переривань від таймеру
        CLR    A        ; (A):=0
        MOV    T,  A    ; (T):=0
        MOV    R5, A   ; (R5):=0
        STRT  T        ; запуск T
L1 2:   JTF    L1 1    ; якщо TF=1, то перехід до L11
        ; установка TF в нуль.
        JMP    L1 2    ;
L1 1:   INC    R 5     ; (R5):=(R5)+1
        MOV    A,  R5  ; (A):=(R5)
        JB3   INT     ; перехід до підпрограми
        ; обслуговування переривань INT,
        ; якщо біт 3 має значення один
        JMP    L1 2    ; перехід якщо біт 3 ≠ 1
INT:    STOP   TCNT    ; зупинити T
        JMP    07h    ; перейти до комірки 7 (вектор
        ; переривань від лічильника подій)

```

ПЕРЕЛІК ЛІТЕРАТУРИ

1. Арифметичні та управляючі пристрої цифрових ЕОМ: Навчальний посібник / В.І.Жабін, І.А.Жуков, І.А.Клименко, Стиренко С.Г. – К.: ВЕК+, 2008. – 176 с.
2. *Бабич М.П., Жуков І.А.* Атестаційні роботи магістрів і спеціалістів: Навчально-методичний посібник. – К. НАУ, 2004. – 216 с.
3. *Жабин В.И.* Архитектура вычислительных систем реального времени. – К.: ВЕК+, 2003. – 176 с.
4. *Жабин В.И., Ткаченко В.В.* Однокристалльные и микропрограммируемые ЭВМ. – К.: Диалектика, 1995. – 116 с.
5. *Жабін В.І., Ткаченко В.В.* Цифрові автомати. Практикум. – К.: ВЕК+, 2004. – 160 с.
6. *Каган Б.М.* Электронные вычислительные машины и системы. – М.: Энергоатомиздат, 1985. – 552 с.
7. *Карцев М.А.* Архитектура цифровых вычислительных машин. – М.: "Наука", 1978. – 295 с.
8. *Микропроцессорные системы: Учебное пособие для вузов / Е.К. Александров, Р.И. Грушевицкий, М.С. Куприянов и др.; Под общ. ред. Д.В. Пузанкова.* – СПб.: Политехника, 2002. – 935 с.
9. *Молчанов А.А., Корнейчук В.И., Тарасенко В.П.* Справочник по микропроцессорным устройствам. – К.: Техніка, 1987. – 288 с.
10. *Новожилов О.П.* Основы микропроцессорной техники / Учебное пособие в двух томах. – М.: ИП РадиоСофт, 2007. – 336 с.
11. *Прикладана теорія цифрових автоматів: Навчальний посібник / В.І.Жабін, І.А.Жуков, І.А.Клименко, В.В.Ткаченко.* – К.: Книжкове видавництво НАУ, 2007. – 364 с.
12. *Проектирование цифровых устройств на однокристалльных микроконтроллерах / В.В. Сташин, А.В. Урусов, О.Ф. Мологонцева.* – М.: Энергоатомиздат, 1990. – 224 с.
13. *Пухальский Г.И., Новосельцева Т.Я.* Цифровые устройства: Учебное пособие для вузов. – СПб.: Политехника, 1996. – 885 с.
14. *Пухальский Г.И.* Проектирование микропроцессорных систем: Учебное пособие для ВУЗов. – СПб.: Политехника, 2001. – 544 с.
15. *Самофалов К.Г., Корнейчук В.И., Тарасенко В.П.* Цифровые ЭВМ. Теория и проектирование. – К.: Вышш.шк. 1989. – 424 с.
16. *Тарабрин В.В., Лунин Л.Ф., Смирнов Ю.Н.* Интегральные микросхемы: Справочник. – М.: Радио и связь, 1990. – 528 с.

17. *Цифровые ЭВМ. Практикум* / К.Г.Самофалов, В.И. Корнейчук, В.П. Тарасенко, В.И.Жабин – К.: Высш.шк. 1989. – 124 с.

МОДЕЛЮЮЧИЙ КОМПЛЕКС *SCM* МК48

SCM (*Single-Chip Machine*) представляє собою систему моделювання роботи внутрішніх компонентів БІС сімейства МК58.

SCM МК48 призначена для:

- моделювання роботи мікроконтролера КР1816ВЕ48 в сукупності з мікросхемою-розширювачем портів вводу виводу КР580ВР43 і блоком зовнішньої пам'яті об'ємом 256 байт;
- розробки і налагодження програм для мікроконтролерів серії МК48;
- дослідження поведінки внутрішніх і зовнішніх сигналів вказаних мікросхем.

Головне вікно програми *SCM* зображено на рис. Б.1. У верхній частині вікна розміщене головне меню програми та інструментальна панель. Опис основних пунктів головного меню та інструментальної панелі дивись у додатку В.

У центральній частині головного вікна розміщуються функціональні пристрої, з яких складається МК48: два послідовні порти *P1* та *P2*, паралельний порт *BUS*, блок управління БУ, таймер та АЛП. Структурі елементи поєднані між собою каналами зв'язку. У структурі кожного елемента МК48 зображені регістри та їх вміст. Під час виконання програми у моделюючому комплексі відображаються зміни вмісту регістрів відповідно до кожного шагу моделювання.

Для завдання вихідних значень у буферах портів необхідно натиснути піктограми (1) (рис. Б.1), розміщені біля кожного з портів. Під час цього відкривається діалогове вікно редагування буферу відповідного порту (рис. Б.2).

Наприклад, для встановлення вихідних значень у буфері порту *P1*, обираємо розділ діалогового вікна «**Буфер для P1**». Натисненням кнопки «**Добавить**», додамо один буфер і встановимо його значення в *C0*. Встановлення значення виконується безпосередньою зміною значень певних бітів. Щоб змінити значення біта на протилежний, необхідно виконати подвійний клік мишкою на певному біті.

Зліва у головному вікні програми відображена структура та вміст пам'яті програм МК48, справа – структура та вміст пам'яті даних.

Single Chip Machine - [Эмуляция работы внутренних компонентов БИС КМ1816ВЕ48]

Файл Работа Опции Настройки Справка

Сброс Шаг Иди к Откат Редактор KP580BP43

Внутреннее ПЗУ

Адрес: 000 Данные: 00

адрес	код	мнемоника
000	00	NOP

Оперативная память

Адрес: 00

R0	00:	00	18:	00	08:	0000
R1	01:	00	19:	00	0A:	0000
R2	02:	00	1A:	00	0C:	0000
R3	03:	00	1B:	00	0E:	0000
R4	04:	00	1C:	00	10:	0000
R5	05:	00	1D:	00	12:	0000
R6	06:	00	1E:	00	14:	0000
R7	07:	00	1F:	00	16:	0000

адрес	+00	+01	+02	+03	+04	+05	+06	+07
00:	00	00	00	00	00	00	00	00
08:	00	00	00	00	00	00	00	00
10:	00	00	00	00	00	00	00	00
18:	00	00	00	00	00	00	00	00
20:	00	00	00	00	00	00	00	00
28:	00	00	00	00	00	00	00	00
30:	00	00	00	00	00	00	00	00
38:	00	00	00	00	00	00	00	00

адрес	+00	+01	+02	+03	+04	+05	+06	+07
000:	00	00	00	00	00	00	00	00
008:	00	00	00	00	00	00	00	00
010:	00	00	00	00	00	00	00	00
018:	00	00	00	00	00	00	00	00
020:	00	00	00	00	00	00	00	00
028:	00	00	00	00	00	00	00	00
030:	00	00	00	00	00	00	00	00
038:	00	00	00	00	00	00	00	00
040:	00	00	00	00	00	00	00	00
048:	00	00	00	00	00	00	00	00
050:	00	00	00	00	00	00	00	00
058:	00	00	00	00	00	00	00	00
060:	00	00	00	00	00	00	00	00
068:	00	00	00	00	00	00	00	00

BusAD

АЛУ

АЛБ В 00 А 00

А7 А6 А5 А4 А3 А2 А1 А0

0 0 0 0 0 0 0 0

Признаки: IE F1 MB PSW C AC F0 RB ... SP ...

0 0 0 08 0 0 0 0 1 0 0 0

Int T0 T1 PR EMA PME ALE W RD

Такт: 0 Фаза: 0 Ожидание указаний Объект КМ1816ВЕ48 готов

Рис. Б.1. Головные вікно програми SCM1

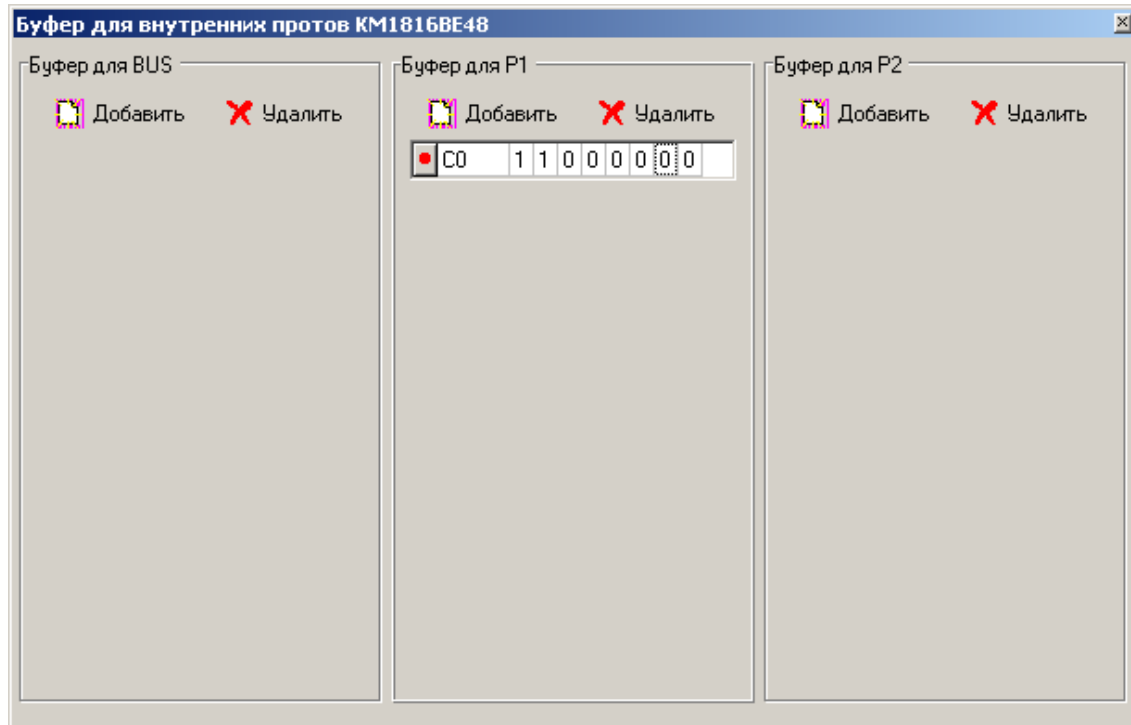


Рис. Б.2. Вікно редагування внутрішніх портів

Для *завантаження вихідного файлу* програми, використовують команду головного меню **Файл** → **Открить**, під час цього відкривається вікно редагування коду програми (рис. Б.3).

Для *збереження результатів редагування* програми необхідно застосувати команду головного меню **Файл** → **Сохранить**. Для компілювання програмами – команду головного меню **Компильця** → **Компильця** (клавіша <CTRL / F9>), для запуску **Компильця** → **Запуск** (клавіша <F9>).

У випадку помилки під час компіляції вказується помилка і номер рядку, в якому вона відбулася. У випадку успішної компіляції, вікно редагування закривається, а в область пам'яті програм головного вікні завантажується вихідна програма. На початку виконання програми встановлюють значення вхідних буферів у вікні редагування портів(рис. Б.2).

Для покрокового *налагодження програми* використовується піктограма інструментальної панелі **Шаг**. Головне вікно програми під час налагодження програми зображено на рис. Б.4.

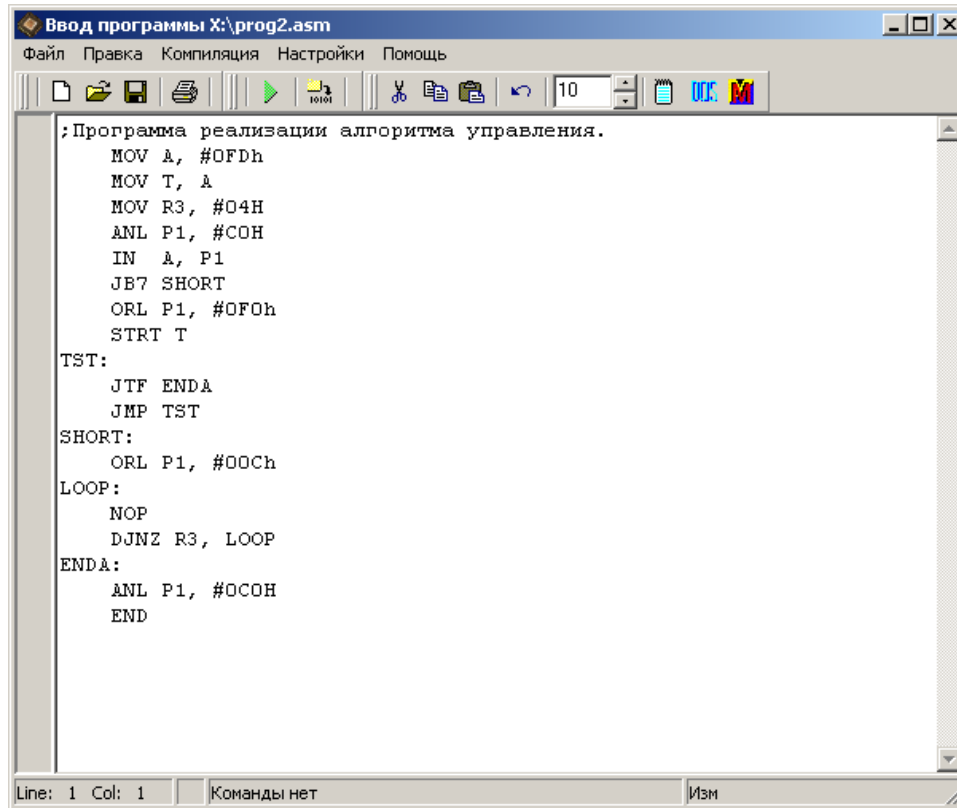
Для виконання програми до першої поміченої команди (або кінця програми), використовується піктограма **Идти к**. Позначка команд виконується кліком лівої кнопки миші на необхідній команді. При цьому обирається необхідна в закладка вікна пам'яті програм – **«Внутреннее ПЗУ»** або **«Внешнее ПЗУ»**.

Зміни, що відбуваються з вмістом регістрів структурних елементів МК48 або комірок пам'яті даних під час покрокового виконання програми помічаються жовтим фоновим кольором (рис. Б.4).

Під час виконання програми можна продивитись *часові діаграми сигналів* пристроїв МК48. За натискання піктограм (2) (рис. Б.1) відкривається вікно часових діаграми відповідного пристрою. Часова діаграма сигналів порту P1 зображена на рис. Б.5. На діаграмі відображаються значення всіх бітів порту в кожний момент часу роботи МК48. Діаграма дозволяє скинути значення бітів, відключивши/включивши живлення, а також виконати побітовий пошук конкретного значення порту на діаграмі.

Вікно часових діаграм сигналів можна також відкрити за допомогою команди головного меню **Работа** → **Открить диаграмму** → **Порта P1** (рис. Б.5).

Для визначення часу сигналу, необхідно виконати клік лівою кнопкою мишки на початку сигналу і правою в кінці. На рис. Б.5 визначена тривалість сигналу – 35 мкс.



```
;Программа реализации алгоритма управления.
MOV A, #0FDh
MOV T, A
MOV R3, #04H
ANL P1, #COH
IN A, P1
JB7 SHORT
ORL P1, #0FOh
STRT T
TST:
JTF ENDA
JMP TST
SHORT:
ORL P1, #00Ch
LOOP:
NOP
DJNZ R3, LOOP
END A:
ANL P1, #0COH
END
```

Line: 1 Col: 1 Команды нет Изм

Рис.Б.3. Вікно редагування програми

Single Chip Machine - [Эмуляция работы внутренних компонентов БИС КМ1816ВЕ48]

Файл Работа Опции Настройки Справка

Сборос Шаг Иतिक Откат Редактор KP580BP43

Внутреннее ПЗУ | Внешнее ПЗУ

Адрес: 013 Данные: 0C

адрес	код	мнемоника
000	23 FD	MOV A,#FDh
002	62	MOV T A
003	BB 04	MOV R3,#04h
005	99 C0	ANL P1,#C0h
007	09	IN A,P1
008	F2 11	JB7 11h
00A	89 F0	ORL P1,#F0h
00C	95	START T
00D	16 16	JTF 16h
00F	04 0D	JMP 00D
011	89 0C	ORL P1,#0Ch
013	00	NOP
014	EB 13	DJNZ R3,13h
016	99 C0	ANL P1,#C0h
018	--	Нет памяти
019	--	Нет памяти
01A	--	Нет памяти
01B	--	Нет памяти
01C	--	Нет памяти
01D	--	Нет памяти
01E	--	Нет памяти
01F	--	Нет памяти
020	--	Нет памяти
021	--	Нет памяти
022	--	Нет памяти
023	--	Нет памяти
024	--	Нет памяти
025	--	Нет памяти
026	--	Нет памяти
027	--	Нет памяти
028	--	Нет памяти
029	--	Нет памяти
02A	--	Нет памяти
02B	--	Нет памяти

Оперативная память

Адрес: 09

Адрес	RB0	Адрес	RB1	Адрес	Stack
R0	00: 00	18: 00	08: 00	08: 0000	
R1	01: 00	19: 00	0A: 00	0000	
R2	02: 00	1A: 00	0C: 00	0000	
R3	03: 04	1B: 00	0E: 00	0000	
R4	04: 00	1C: 00	10: 00	0000	
R5	05: 00	1D: 00	12: 00	0000	
R6	06: 00	1E: 00	14: 00	0000	
R7	07: 00	1F: 00	16: 00	0000	

адрес	+00	+01	+02	+03	+04	+05	+06	+07
00:	00	00	00	04	00	00	00	00
08:	00	00	00	00	00	00	00	00
10:	00	00	00	00	00	00	00	00
18:	00	00	00	00	00	00	00	00
20:	00	00	00	00	00	00	00	00
28:	00	00	00	00	00	00	00	00
30:	00	00	00	00	00	00	00	00
38:	00	00	00	00	00	00	00	00

адрес	+00	+01	+02	+03	+04	+05	+06	+07
000:	00	00	00	00	00	00	00	00
008:	00	00	00	00	00	00	00	00
010:	00	00	00	00	00	00	00	00
018:	00	00	00	00	00	00	00	00
020:	00	00	00	00	00	00	00	00
028:	00	00	00	00	00	00	00	00
030:	00	00	00	00	00	00	00	00
038:	00	00	00	00	00	00	00	00
040:	00	00	00	00	00	00	00	00
048:	00	00	00	00	00	00	00	00
050:	00	00	00	00	00	00	00	00
058:	00	00	00	00	00	00	00	00
060:	00	00	00	00	00	00	00	00
068:	00	00	00	00	00	00	00	00

БУ

PC 013 IP 013 IR 10001001

Таймер

TCNT FD TF 0 TIE 0

АЛУ

АЛБ В 0C A C0

А7 А6 А5 А4 А3 А2 А1 А0

1 1 0 0 0 0 0 0

Признаки: IE F1 MB PSW C AC F0 RB SP

0 0 0 0 0 0 1 0 0 0

Int T0 T1 PR EMA PME ALE W RD

Такт: 66 Фаза: 1 Ожидание указаний Объект КМ1816ВЕ48 готов

Рис. Б.4. Вікно налагодження програми

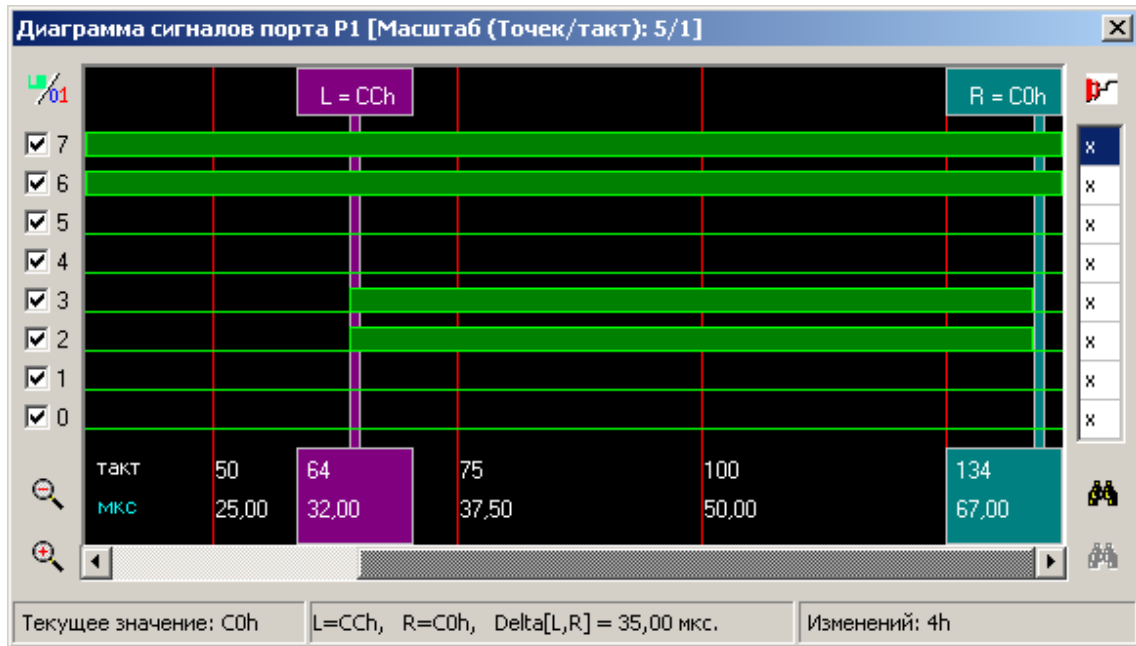


Рис. Б.5. Вікно перегляду часових діаграм сигналів