Ужгородський національний університет
Інженерно-технічний факультет
Кафедра приладобудування

ЦИФРОВА ЕЛЕКТРОНІКА

ЛАБОРАТОРНИЙ ПРАКТИКУМ

УЖГОРОД - 2024

Чиура І.І.Петраченков О.Є

ЦИФРОВА ЕЛЕКТРОНІКА. Лабораторний практикум. Ужгород. Видавництво Ужгородського національного університету. 2024, -38 с.

В методичній розробці описані лабораторні роботи з цифрової електроніки та методика їх виконання. Розглядаються цифрові вузли і пристрої, побудовані на цифрових інтегральних мікросхемах серій ТТЛ, а також способи їх практичного використання. Основна увага приділена вивченню алгоритмів роботи комбінаційних та послідовнісних цифрових автоматів, проектуванню цифрових пристроїв на основі інтегральних мікросхем середнього ступеня інтеграції.

Для студентів інженерно-технічних факультетів вищих навчальних закладів, які навчаються за напрямком: автоматизація та комп'ютерно-інтегровані технології,.

Рецензент:
кандидат фіз. - мат. наук, доцент Спесивих О.О. ,

Рекомендовано до друку методичною радою інженерно-технічного факультету (протокол № від січня 2024 р.)

З М І С Т

ПЕРЕДМОВА

Мікропроцесори і мікро-ЕОМ стали складовою частиною надзвичайно різноманітних пристроїв, приладів та систем обробки інформації. Завдяки можливості програмного керування обробкою інформації мікропроцесорам притаманні властивості універсальних керуючих пристроїв цифрових систем.

Оскільки, реалізація задач керування технологічними процесами і об'єктами з використанням мікропроцесорів і мікро-ЕОМ здійснюється шляхом програмування процесів накопичення та обробки даних, діагностики, формування та виведення сигналів керування і ін., тому розробка, виготовлення і експлуатація машин, обладнання та систем, виготовлених на основі мікропроцесорної техніки, вимагає відповідної підготовки студентів вузів.

Лабораторний практикум, як і курс "Основи мікропроцесорної техніки" забезпечує перший рівень підготовки спеціалістів і вирішує задачі формування у студентів знань з архітектури мікропроцесорів і мікро-ЕОМ, з принципів обробки і передачі інформації в мікропроцесорних системах, а також навиків їх програмування.

Посібник містить сім лабораторних робіт. Чотири перших роботи орієнтовані на вивчення системи команд однокристального мікропроцесора Intel 8080, принципів обробки інформації цим процесором та основних правил написання програм на асемблерному рівні. П'ята лабораторна робота присвячена вивченню способів обміну інформацією в мікропроцесорних система на прикладі типових інтерфейсів. При виконані шостої роботи студенти знайомляться з структурою і типовим програмним забезпеченням (драйвери зовнішніх пристроїв, діалогові програми керування) мікропроцесорної вимірювальної системи. В останній роботі вивчаються стандарти та методи програмування системи модульної електроніки КАМАК.

В додатках приведений довідковий матеріал, який в сукупності з теоретичними відомостями до кожної роботи забезпечує необхідну інформацію для вивчення і практичної розробки програмних і апаратних засобів мікропроцесорних пристроїв.

Апаратною основою лабораторного практикуму є типові мікропроцесорні пристрої виготовлені на базі мікропроцесорного комплекту серії КР580.

Навчальний посібник складений на основі досвіду проведення лабораторних робіт з мікропроцесорної техніки на кафедрах фізики напівпровідників та промислової електорніки Ужгородського держуніверситету. Він розрахований на студентів фізичних факультетів університетів. Проте може бути використаний студентами інших спеціальностей. Посібник може стати у пригоді викладачам вузів, які займаються постановкою лабораторних робот по відповідній тематиці.

Автори вдячні колегам по кафедрі, дискусії з якими сприяли виробленню структури посібника і його написанню, колективу ТОВ “Офіс-сервіс” за постійну матеріально-технічну допомогу при організації практикуму та надання навчальних і системних програм.

Розділ другий

ЛАБОРАТОРНІ РОБОТИ

Лабораторна робота №1

Ознайомлення з роботою на навчальній мікро-ЕОМ

Мета роботи. Ознайомитися з структурою навчального мікропроцесорного комплекту (УМК), картою пам'яті, органами керування та режимами роботи навчальної мікро-ЕОМ.

Короткі відомості з теорії

Призначення УМК, його структура, карта пам'яті мікро-ЕОМ описані в першому розділі. Тут розглянемо структуру типового 8-розрядного мікропроцесора.

На мал. 1.1 та 1.2 приведені умовне графічне позначення та структурна схема мікропроцесора серії К580, на основі якого побудована навчальна мікро-ЕОМ. Велика інтегральна схема КР580ВМ80А (аналог Intel 8080A) представляє собою 8-розрядний процесор з фіксованим набором команд, в якому суміщені операційний та керуючий вузли. Призначення виводів мікропроцесора приведено в табл. 1.1.

Регістри мікропроцесора. Для зберігання даних, які використовуються в операціях, передбачено сім 8-розрядних регістрів. Регістр А, що зветься акумулятором, служить для обміну інформацією із зовнішніми пристроями (ПЗПр, ОЗПр, ПрУВ та ін. на мал. 0.1), тобто вміст цього регістру може бути переданий у зовнішній пристрій, або із зовнішнього пристрою в нього може бути прийняте число. При виконані арифметичних, логічних операцій та операцій переміщення він служить джерелом операнда (числа, яке приймає участь в операції). В акумулятор також пересилається результат виконання операції в арифметико-логічному блоці. Шість інших регістрів, що позначаються B, C, D, E, H, L утворюють блок регістрів загального призначення (РЗП). Ці регістри можуть використовуватись для зберігання як даних, так і адрес. Якщо в них зберігаються дані,

то вони використовуються як окремі, 8-розрядні регістри. В тих випадках, коли потрібно зберігати 16-розрядні слова, то РЗП об’єднують в пари BC, DE, HL.

Показник стеку SP (16-розрядний) служить для адресації особливого виду пам’яті - стеку.

Лічильник команд PC (16-розрядний) використовується для зберігання адреси команди. Після вибору із пам’яті чергової команди вміст PC збільшується на одиницю і, таким чином, формується адреса наступної команди (при умові, якщо не виконуються команди умовних і безумовних переходів).

При звертанні до пам’яті джерелом адреси може бути вміст любої пари BC, DE, HL.

При виведені коду адреси з Мъэучэъснсэчэсэъсз u▪ D▪ UЃ[Ѓ] Сигнал уведення даних в мікропроцесор Сигнал виведення даних із мікропроцесора Сигнал синхронізації Напруга живлення +5 В Підтвердження захоплення шин Сигнал ГОТОВНІСТЬ від зовнішнього пристрою Сигнал ОЧІКУВАННЯ для зовнішнього пристрою Напруга живлення +12 В

Арифметико-логічний блок (АЛБ). У 8-розрядному АЛБ передбачено можливість виконання: арифметичних операцій додавання і віднімання чисел, які записані двійковим кодом (від’ємні числа записуються у доповняльному коді); логічних операцій та арифметичних операцій над десятковими числами, які представлені в двійково-десятковому коді.

Регістр ознак, або індикатор F служить для зберігання інформації про ознаки, які характеризують результат, одержаний в АЛБ, після виконання чергової операції. Ці ознаки представлені наступними бітами:

- біт перенесення C;
- біт парності P;
- біт додаткового перенесення AC;
- біт нульової ознаки (біт нуля) Z;
- біт знаку S.

Не при всіх операціях, які виконуються в АЛБ, встановлюються індикатори.

Керуючий блок складається із регістру команд, куди заноситься перший байт команди, та схеми керування і синхронізації, яка містить мікропрограми окремих операцій.

Буфери шини даних і шини адреси забезпечують зв’язок мікропроцесора з зовнішніми шинами даних і адреси. Особливістю буферних схем є те, що в кожному розряді використовуються логічні елементи з Z-станом. Це дає можливість процесору відключатися від зовнішніх шин і передавати їх у користування зовнішнім пристроям.

Режими роботи мікропроцесора

Мікропроцесор - це цифровий автомат, який виконує операції з цифровою інформацією у відповідності з програмою, яка зберігається у пам'яті. Тому робота по програмі є основним режимом роботи.

Після початкового запуску, коли на вхід RESET мікропроцесора подається сигнал лог. "1" в лічильнику команд PC встановлюється число 0000H. Після перепаду сигналу RESET з високого рівня на низький (з лог. "1" на лог. "0") мікропроцесор виводить на ША код адреси 0000H і читає інформацію з комірки пам'яті, яка розміщена по цій адресі. Вміст комірки розшифровується в блоці управління як код операції і мікропроцесор розпочинає виконувати команди у відповідності з програмою.

В процесі виконання програми МП може перейти в режим ОЧІКУВАННЯ (на вході READY лог. "0"). В цьому режимі припиняється процес обробки даних, формується сигнал високого рівня (лог. "1") на виході WAIT, який підтверджує цей режим. В режимі ОЧІКУВАННЯ мікропроцесор може знаходитися як завгодно довго, поки сигнал на вході READY не прийме значення лог. "1", після цього виконання програми продовжується.

Режим ПЕРЕРИВАННЯ - це такий режим роботи МП коли по сигналу від зовнішнього пристрою мікропроцесор переходить на виконання спеціальної підпрограми (підпрограми обробки переривання).

Ще один режим роботи МП - режим ЗАХОПЛЕННЯ (зовнішнім пристроєм) шин даних і адреси. Стан ЗАХОПЛЕННЯ характеризується тим, що мікропроцесор, одержавши запит на захоплення по входу HOLD, закінчує виконання чергової команди і переводить буферні схеми шин даних і адреси в Z-стан. При цьому МП відключається від зовнішніх шин, передає їх у користування пристрою, який подав запит на захоплення, і зупиняється. По закінчені дії сигналу HOLD МП продовжує виконання програми з місця, де було зупинено її виконання.

Зупинити роботу мікропроцесора можна і програмним шляхом з допомогою спеціальної команди зупинки HLT (код операції 76H). Ця команда зупиняє виконання програми і переводить МП в режим ЗУПИНКИ - буфери даних і адреси переходять у Z-стан, МП відключається від зовнішніх шин, а на виході WAIT встановлюється лог. "1". З режиму ЗУПИНКИ мікропроцесор може бути виведений сигналом перезапуску RESET, або сигналом запиту на переривання по входу INT.

Завдання для домашньої підготовки

1. Ознайомитися з описом навчальної мікро-ЕОМ.

2. Ознайомитися з типовою мінімальною структурою мікро-ЕОМ, методами організації шин, підключенням пам’яті і зовнішніх пристроїв до магістралей МП.

3. Вивчити основні режими роботи управляючої програми Монітор.

4. Вивчити внутрішні регістри МП КР580ВМ80А.

Завдання до лабораторної роботи

Завдання 1. Дослідити порядок вмикання мікро-ЕОМ. Порядок виконання завдання:

- натиснути кнопку "~";

- натиснути кнопку "СБ". Спостерігати свічення знаку "–" зліва на дисплеї;

Завдання 2. Дослідити вміст пам’яті. Порядок виконання завдання:

- натиснути кнопку "П". При цьому повинен погаснути знак "-" на дисплеї;

- послідовно натиснути кнопки"0", "В", "Е", "7". Переконатися, що при цьому кожна цифра буде записана в молодший розряд адресної частини дисплея і відбувається одночасне переміщення всіх знаків на адресному дисплеї на один знак уліво;

- після уведення четвертої цифри натиснути кнопку "|__|". На дисплеї повинно появитися число 21Н ;

- натиснути кнопку "|__|". Мікро-ЕОМ збільшить на одиницю адресу на адресній частині дисплея і виведе на дисплей вміст комірки пам’яті по цій адресі. Послідовно натискуючи кнопку "|__|" перевірити вміст пам’яті ОЗПр;

- натиснути кнопку "ВП";

- натиснути кнопку "П", увести код 0000Н і натиснути кнопку"|__|".

На дисплеї повинно появитись число С3Н. Переглянути вміст декількох комірок пам’яті ПЗПр (в перших трьох комірках пам’яті зберігається код команди безумовного переходу по адресі 0040Н - JMP 0040Н. Перший байт С3 - це код операції, а два послідуючих байти - адреса).

Завдання 3. Запис даних у пам’ять мікро-ЕОМ. Порядок виконання завдання:

- натиснути кнопку "П", набрати адресу 0800Н і натиснути кнопку "|__|". На дисплеї появиться вміст комірки пам’яті по цій адресі;

- натиснути кнопки "1","2". Записати число 12Н в комірку з адресою 0800Н, натиснувши кнопку "|__|". При цьому на дисплеї появиться адреса 0801Н і вміст комірки пам'яті по цій адресі;

- послідовно набираючи дані (дві любі цифри від 0 до F) і натискуючи кнопку "|__|" записати дані в пам'ять ЕОМ;

- закінчити директиву читання і модифікації пам'яті натиснувши кнопку "ВП";

- набрати адресу 0800Н і переконатись, що уведені в мікро-ЕОМ дані записані в оперативну пам'ять.

Завдання 4. Запис чисел в програмно-доступні регістри мікропроцесора. Порядок виконання завдання:

- натиснути кнопку "СБ";

- натиснути кнопку "РГ", а потім одну із кнопок ідентифікаторів регістрів. Ідентифікаторами регістрів є символи, які визначають регістри мікропроцесора:

A - регістр A; **L** - регістр L;

B - регістр B; **F** - регістр ознак (індикатор);

C - регістр C; **SL** - молодший байт показника стеку;

D - регістр D; **SH** - старший байт показника стеку;

E - регістр E; **PL** - молодший байт лічильника команд;

H - регістр H; **PH** - старший байт лічильника команд.

Відповіддю ЕОМ на уведення ідентифікатора є індикація на дисплеї вмісту вибраного регістра в Н-кодах.

- для запису числа у вибраний регістр уведіть нове число з допомогою інформаційних кнопок і натисніть кнопку "|__|";

- директива читання і модифікації вмісту регістрів мікропроцесора закінчується натисненням на кнопку "ВП".

Завдання 5. Запуск демонстраційної програми. Порядок виконання завдання:

- натисніть кнопку "СТ";

- уведіть адресу 0400Н. Ця адреса є адресою початку демонстраційної програми "Переміщення", яка записана в ПЗПр;

- натисніть кнопку "ВП". Спостерігайте на дисплеї переміщення чисел 1, 2, 3, 4. Переконатися, що виконання програми може бути зупинене кнопкою "ПР". При зупинці програми на дисплей виводиться адреса, записана у лічильнику команд;

- прочитати вміст регістрів мікропроцесора на момент зупинки програми.

Зміст звіту

Звіт за лабораторну роботу повинен містити:

- назву та мету роботи;
- структурну схему навчальної мікро-ЕОМ та її опис;
- карту пам’яті мікро-ЕОМ;
- інформацію про вміст внутрішніх програмно-доступних регістрів мікропроцесора після перезапуску мікро-ЕОМ та після переривання демонстраційної програми;
- порядок виконання додаткового завдання та одержані результати.

Додаткові завдання

Д1.1. Підрахувати контрольну суму масиву пам’яті з адреси 0005Н по 0020Н. Пояснити результат.

Д1.2. Дослідити сигнали на шинах даних і адреси при по-кроковому виконані програми з адреси 0457Н.

Д1.3. Дезасемблерувати (записати на мові Асемблера) програму з адреси 035ВН по 0364Н.

Д1.4. Ознайомитися з роботою навчальної мікро-ЕОМ типу "Електроніка 580".

Д1.5. Розробити схему формування магістралі керування для мікро-ЕОМ на основі МП КР580ВМ80А.

Д1.6. Розробити схему запису слова стану мікропроцесора в ре3

10. Які регістри МП називають програмно-доступними?

11. Яке призначення регістра ознаків мікропроцесора?

12. Назвіть і поясніть призначення кнопок на пульті керуванняння мікро-ЕОМ.

13. Яка інформація записується у регістри мікропроцесора і ОЗПр в процесі виконання програми початкового запуску мікро-ЕОМ.

Лабораторна робота № 2

Написання та виконання простих програм

Мета роботи. Дослідження виконання мікропроцесором окремих команд і простих програм; дослідження різних методів адресації в програмах; написання програм на мові Асемблера.

Короткі відомості з теорії

Цикли роботи мікропроцесора. Мікропроцесор КР580ВМ80А має фіксований набір команд. Час виконання команди визначається процесом одержання, розшифровування та виконання команди. Цей час складається із ряду часових інтервалів. Найбільш короткий інтервал часу, рівний періоду синхросигналів МП, називається машинним тактом.

Такт - найменший проміжок часу, необхідний для виконання однієї елементарної дії в МП. Цю елементарну дію називають мікрооперацією. Такт рівний періоду Т синхросигналів F1 і F2, які мають однакові періоди (мал. 2.1.), проте тривалості імпульсів t1, t2 різні. Для навчальної мікро-ЕОМ типу УМК Т=450 нс. На протязі одного машинного такту МП не змінює свого стану. В кожному стані МП може знаходитися протягом певного числа тактів. Але є три стани, в яких МП може перебувати необмежено довго, лише ціле число тактів. Це стани - ОЧІКУВАННЯ, захоплення та зупинки.

Час, необхідний для вибору 1 байта інформації із пам’яті, або із зовнішнього пристрою, або для виконання команди, формат якої складає одне машинне слово, називається машинним циклом. Машинний цикл для МП може складатися із 3-5 машинних тактів. В залежності від типу команди час її виконання може становити 1-5 машинних циклів. Для мікропроцесора серії К580 існує 10 різних типів машинних циклів: вибір коду команди (цикл М1), читання даних із пам’яті, запис даних у пам’ять, вибір даних із стеку, запис даних у стек, уведення даних із зовнішнього пристрою, запис даних у зовнішній пристрій, цикл обслуговування переривання, зупинка, обслуговування переривання в режимі зупинки. Першим машинним циклом при виборі любої команди - є цикл М1.

Таблиця 2.1. Значення розрядів слова стану процесора для різних машинних циклів

Тип машинного циклуD7D6D5D4D3D2D1D0Вибір команди (M1)10100010Читання з пам’яті10000010Запис у пам’ять00000000Читання з стеку10000110Запис у стек00000100Уведення із зовнішнього пристрою01000010Виведення у зовнішній пристрій00010000Підтвердження дозволу на перери-вання00100011Підтвердження зупинки10001010Підтвердження переривання при зупинці00101011

На початку кожного машинного циклу (мал. 2.2) МП виставляє на шину даних вісім біт інформації (слово стану процесора), яка характеризує стан його внутрішніх вузлів і вказує, які дії буде виконувати МП підчас даного машинного циклу. Ця інформація знаходиться на шині даних протягом дії сигналу SYNC, який завжди появляється на першому такті любого машинного циклу (табл. 2.1).

Призначення розрядів слова стану процесора: DO(INTA) - підтвердження запиту на переривання; D1(WO) - запис у пам’ять, або виведення у зовнішній пристрій; D2(STAK) - операція зі стеком; D3(HLTA) - підтвердження зупинки після виконання команди HLT; D4(OUT) - виведення у зовнішній пристрій; D5(M1) - перший цикл команди; D6(INP) - уведення із зовнішнього пристрою; D7(MEMR) - читання із пам’яті.

На кожному машинному циклі мікропроцесор перевіряє стан сигналу на вході READY. Лог. "0" на цьому вході зупиняє нормальну роботу МП, при цьому на ШД і ША знаходиться вся інформація, яка передається в даному циклі. В навчальній мікро-ЕОМ це використовується для дослідження виконання команд по машинних циклах.

Програмування мікропроцесора КР580ВМ80А. МП здатний сприймати лише програми, які складаються із послідовності команд, що представлені двійковими кодами. Програмування безпосередньо в машинних кодах веде до постійного використання багаторозрядних двійкових чисел, які є як кодами команд, так і кодами операндів.

Розглянемо програму додавання двох чисел, яка записана в машинних двійкових кодах. Одне із чисел 01100100 знаходиться в регістрі загального призначення B, а друге - 00010111 розміщене в комірці пам’яті з адресою 0000101100000000. Фрагмент програми приведений в табл. 2.2.

Таблиця 2.2. Програма додавання двійкових чисел 01100100 і 00010111 в машинних кодах

Адреса комірки пам’ятіВміст комірки пам’яті0000 1000 0000 00000111 10000000 1000 0000 00010010 00010000 1000 0000 00100000 00000000 1000 0000 00110000 10110000 1000 0000

01001000 01100000 1000 0000 01010111 0110 Перша команда - пересилка вмісту регістра B в акумулятор є однобайтовою, має код операції 01111000. Код операції розміщений ОЗПр по адресі 0000100000000000, яка для даної програми є початковою. Наступна команда - трьохбайтова, має код операції 00100001, який є першим байтом команди. Другий і третій байти, що розміщені в наступних комірках пам'яті і представляють собою молодшу і старшу частини адреси 0000101100000000, передаються по цій команді в регістри L і H відповідно. Остання команда має код 01110110 і являється командою зупинки (HLT).

В результаті виконання програми в акумуляторі буде записане число 0111101.

Запис програми, приведений в табл. 2.2, є громіздким і незручним. Тому прийнято машинні коди записувати у шістнадцяткові (або у вісімкові) системах числення.

В табл. 2.3 приведена таж сама програма , але записана в шістнадцяткових кодах. Поряд з кодом операції приведені мнемокоди команд і назви операндів, які прийняті в системі команд мікропроцесора Intel 8080. Наприклад, шістнадцятковий код команди 78H, що має мнемонічне позначення MOV A, B, відповідає операції пересилки вмісту регістру B у акумулятор A.

Таблиця 2.3. Програма додавання чисел
64H і 17H в шістнадцядкових кодах

МнемокодАдресаВміст пам'ятіMOV A, B080078LXI H, 0B000801210802000803 0BADD M080486HLT080576 Ефективність програмування в машинних кодах і мнемокодах невисока, а імовірність появи помилки різко зростає при збільшені довжини програми. Основні труднощі, які зустрічаються при написані програм в мнемокодах, - необхідність переведення мнемокодів команд в шістнадцяткову форму і розподіл комірок пам'яті по абсолютних адресах. Написання програм значно спрощується при використані мови програмування Асемблера.

Мова Асемблера спрощує написання команд, полегшує пошук помилок в програмах, дозволяє легко вносити зміни в написані програми. У мові Асемблера замість машинних кодів використовується мнемонічна форма запису операцій, які виконує мікропроцесор. В мнемонічній формі (у вигляді сполучення букв, взятих із відповідних англійських слів) представляють вид операції, яку виконує процесор, операнди та адреси. Кожній команді на мові Асемблера відповідає команда в машинних кодах.

Перед виконанням програму необхідно перекласти з мови Асемблера на мову кодових комбінацій і в такому вигляді розмістити в пам'яті мікропроцесорного пристрою.

Такий переклад здійснюється в мікро-ЕОМ з допомогою програми трансляції, яка називається Асемблером. Мовою Асемблера можна користуватися для програмування і в тих випадках, коли відсутня програма для трансляції. Трансляція у цьому випадку здійснюється вручну, користуючись таблицями команд (така процедура називається ручним асемблюванням).

Мова Асемблера індивідуальна для кожного мікропроцесорного комплекту, тобто кожний мікропроцесорний комплект має свою мову Асемблера, яка відмінна від мови Асемблера інших комплектів.

Наступний рівень мови програмування - мова Макроасемблера. В ній передбачена можливість присвоювати ім’я певній послідовності команд. Тоді в любому місці програми, де повинна бути ця послідовність, вказується лише ім’я послідовності. Використання мови Макроасемблера скорочує запис програм (в середньому на 5...20%).

Наступний рівень мови програмування - мова високого рівня. Мови високого рівня наближаються до звичайної математичної мови, яка описує процес розв’язування задачі, і тому вони легко засвоюються.

Мова Асемблера. Програма на мові Асемблера представляється у вигляді послідовності речень, кожне з яких займає окремий рядок.

Кожне речення на мові Асемблера містить чотири частини (полів): поле мітки, поле коду операції, поле операнда і поле коментарів.

1. *Поле мітки*. Якщо реченню присвоюється ім’я, то воно записується в полі мітки і після імені ставиться двокрапка. Мітка утворюється з латинських букв в любій послідовності (мова Асемблера використовує лише великі букви латинського алфавіту) і цифр, причому першим символом мітки повинна бути буква. Мітка повинна містити не більше п’яти символів. В одній програмі не може бути дві однакові мітки. В машинній мові мітці відповідають адреси комірок пам’яті, які містять команди, до яких передбачається звертатися в процесі виконання програми.

2. *Поле коду операції.* Це поле містить мнемокод команди або псевдокоманди.

3. *Поле операнда*. В полі операнда записується додаткова інформація, необхідна для виконання операції. Операндами можуть бути:

а) ідентифікатори регістрів (A, B, C, D, E, H, L, M);

б) ідентифікатори регістрових пар (B, D, H), акумулятора і регістра признаків - PSW, лічильника команд - PC, показника стеку - SP;

в) безпосередні дані, представлені у двійкові, шістнадцяткові, вісімкові або десяткові системах числення. Для позначення системи числення після константи пишеться одна із латинських літер: B (двійкова), Q (вісімкова), H (шістнадцяткова). Десяткові константи записуються у звичайному вигляді. Константа повинна розпочинатися з цифри. Якщо першою цифрою є шістнадцяткові - A, B, C, D, E, F, то перед ними добавляється ноль. Константа може задаватися в коді ASCII (KOI-8). В цьому випадку вказуються один або два символи, обмежені одинарними лапками;

г) мітка, яка вказує адресу операнда;

д) вираз, який складається із констант, сполучених арифметичними або логічними операторами.

4. *Поле коментарів*. Розпочинається символом ";" (крапка з комою). Воно служить для запису любих пояснень змісту виконуваних дій, котрі змогли б полегшити читання програми. Під коментар можна виділити повні рядки, розпочавши їх символом ";". Записи в полі коментарів потрібні лише програмісту, при трансляції вони ігноруються Асемблером.

Псевдо команди Асемблера. Команди програми визначають дії, які виконуються мікропроцесором над даними в процесі розв'язування задачі. Псевдокоманди не зв'язані з діями мікропроцесора над даними, вони лише повідомляють необхідні Асемблеру відомості і використовуються лише в процесі трансляції програми на мову машинних кодів. В табл. 2.4 приведені найбільш вживані псевдокоманди.

Таблиця 2.4. Псевдокоманди Асемблера

Ім'я псевдокомандиПриклад записуМіткаКодОперандКоментарORG EQU DB DW DS END ORG 0200H ; Команди розміщуються, почи- ; наючи з адреси 0200H. TIME EQU 56 ; Константі з ім'ям TIME ; присвоїти значення 56. PR: DB 37 ; Для однобайтової змінної PR ; резервується комірка в яку ; заноситься значення 37. INIT: DW 0 ; Для двобайтової змінної INIT ; резервується дві комірки в ; які заноситься 0. ARR: DS 100 ; Для масиву з ім'ям ARR ; резервується 100 комірок в ; пам'яті. END ; Кінець програми.

Розглянемо програму (програма 2.1), яка завантажує в акумулятор число з комірки з адресою 0B00H, інвертує його і результат заносить в комірку з адресою 0B01H.

Програма 2.1 (в мнемокодах)

Мнемокод Коментар

LDA 0B00H ; одержати число з адреси 0B00H

```
CMA          ; інвертувати число
STA 0B01H    ; записати результат по адресі 0B01H
RST1         ; перервати виконання програми
```

Для запису програми в пам'ять мікро-ЕОМ необхідно перевести мнемокоди команд в машинні коди. Команди в програмі можуть бути одно-, дво-, або трьохбайтні і повинні в пам'яті займати відповідно одну, дві або три адреси.

Програма 2.1 (розміщення по адресах пам'яті)

Адреса	Число	Коментар
0800	3A	; код команди LDA
0801	00	; молодший байт адреси
0802	OB	; старший байт адреси
0803	2F	; код команди CMA
0804	32	; код команди STA
0805	01	; молодший байт адреси
0806	0B	; старший байт адреси
0807	CF	; код команди RST1

Попередній запис програм зручно проводити в більш компактні формі. В програмі вказується лише початкова адреса кожної команди. При цьому враховується, що в залежності від формату, команди в пам'яті будуть займати від однієї до трьох послідовних комірок. При такому запису в лівому стовпці вказується лише адреса команд в програмі.

Програма 2.1 (загальний вигляд запису)

Адреса	Машинний код	Мітка	Мнемокод	Коментар
0800	3A 00 0B		LDA, 0B00H	; одержати число
0803	2F		CMA	; інвертувати число
0804	32 01 0B		STA, 0B01H	; записати по адресі 0B01
0807	CF		RST1	; перервати виконання ; програми

В програмі 2.1 використовується прямий спосіб адресації.

Розглянемо аналогічну програму з використанням непрямого способу адресації (програма 2.2).

Програма 2.2.

Адреса	Машинний код	Мітка	Мнемокод	Коментар
0800	21 00 0B		LXI H, 0B00H	; записати в регістри H, L ; число 0B00H
0803	7E		MOV A, M	; одержати число з ; адреси, яка вказана в ; регістрах H, L
0804	2F		CMA	; інвертувати число в ; акумуляторі
0805	23		INX H	; збільшити на 1 число ; в регістрах H, L
0806	77		MOV M, A	; записати число із ; акумулятора по адресі, яка ; вказана в H, L
0807	CF		RST1	; перервати виконання ; програми

Завдання для домашньої підготовки

1. Ознайомтесь з структурою команд мікропроцесора КР580ВМ80А.

2. Вивчіть методи програмування на мові Асемблера і в машинних кодах для МП КР580ВМ80А.

3. Розгляньте правила виконання команд INR A (3C), DCR A (3D), ADD A (87), ANA A (A7), ORA A (B7), CMP A (BF) (в дужках вказані машинні коди команд в шістнадцяткові системі числення).

4. Проаналізуйте результати виконання програми 2.1, якщо по адресі 0803H будуть записані коди команд, які приведені в п.3 завдання. Результати виконання програми для різних команд занести в таблицю 2.5.

Таблиця 2.5.

Число, записане по адресі 0B00HКоманда, записана по адресі 0803HЧисло, записане по адресі 0B01H

5. Розробіть програми: а) збільшення на 5 числа, яке записане по адресі 0B00H, і запису результату по адресі 0B01H (програма 2.3); додавання чисел, які записані по адресах 0B00H ТА 0BA0H, і запису результату по адресі 0B01H (програма 2.4).

Завдання до лабораторної роботи

Завдання 1. Дослідити програму 2.1. Порядок виконання завдання:

- увести в мікро-ЕОМ програму 2.1;

- записати по адресі 0B00H досліджуване число;

- здійснити пуск програми з адреси 0800H. Перевірити результат виконання програми, прочитавши число по адресі 0B01H;

- дослідити процес виконання програми по командах;

- дослідити процес виконання програми по машинних циклах;

- замінюючи в програмі 2.1 команду CMA на команди, приведені в п.3 завдання для домашньої підготовки, дослідити результат виконання програми по числу, записаному по адресі 0B01H. Перевірити табл. 2.5.

Завдання 2. Дослідити програму 2.2. Порядок виконання завдання:

- увести в мікро-ЕОМ програму;

- записати по адресі 0B00H досліджуване число;

- здійснити пуск програми з адреси 0800H. Перевірити результат виконання програми по числу, яке записане по адресі 0B01H;

- дослідити процес виконання команди MOV A, M по машинних циклах.

Завдання 3. Дослідити програму 2.3. Порядок виконання завдання:

- увести в мікро-ЕОМ програму;

- здійснити її пуск і перевірити результат її виконання для чисел 05, 0FEH, записаних по адресі 0B00H.

Завдання 4. Дослідити програму 2.4. Порядок виконання завдання:

- увести в мікро-ЕОМ програму 2.4;

- перевірити результат виконання програми для чисел 0BH і 0B0H, 0FEH і 0B5H.

Зміст звіту

Звіт повинен містити:

- назву та мету роботи;

- заповнену таблицю 2.5 для випадків виконання програми 2.1 при використані команд, приведених в п.3 завдання для домашньої підготовки;

- розроблені при підготовці до лабораторної роботи програми 2.3, 2.4;

- результати дослідження роботи програм для завдань 1 - 4;

- порядок виконання додаткового завдання та одержані результати.

Додаткові завдання

Д2.1. Зобразіть часові діаграми процесу виконання мікропроцесором команд: CMA; MOV A, M; STA ADDR (тут і надалі завдання сформульовані стосовно мікропроцесора КР580ВМ80А).

Д2.2. Зобразіть часові діаграми процесу виконання МП команд: MVI A, DATA; LHLD ADDR.

Д2.3. Поясніть роботу МП і намалюйте часові діаграми його роботи в режимі ПЕРЕРИВАННЯ.

Д2.4. Поясніть роботу МП і намалюйте часові діаграми в режимах ОЧІКУВАННЯ і ЗАХОПЛЕННЯ.

Д2.5. Оцініть час виконання програми 2.2 навчальною мікро-ЕОМ.

Д2.6. Напишіть програму порівняння чисел, які записані по адресах 0B00H і 0B01H, та запису більшого із них в регістр B (програма 2.5).

Д2.7. Напишіть програму додавання двох 16-ти розрядних чисел, які записані по адресах 0B00H, 0B01H і 0B02H, 0B03H (молодші і старші байти чисел, відповідно) та запису результату по адресі 0B04H та 0B05H (програма 2.6).

Д2.8. Користуючись технічним описом навчального мікропроцесорного комплекту УМК поясніть, які дії виконує мікро-ЕОМ по команді RST1.

Контрольні запитання

1. Які проміжки часу називають машинним тактом? машинним циклом?
2. Назвіть машинні цикли мікропроцесора КР580ВМ80А.
3. Як підрахувати час виконання команди мікропроцесором?
4. Які основні типи команд мікропроцесора КР580ВМ80А Ви знаєте?
5. Поясніть правила виконання команд ADD A, ADC B, SUB A.
7. Поясніть правила виконання команди RST 1.
8. При яких командах мікропроцесора змінюється вміст його регістра SP?
9. Поясніть часові діаграми на мал. 2.1
10. Поясніть часові діаграми на мал. 2.2.
11. Поясніть часові діаграми на мал. 2.3.
12. Спроектуйте схему запису слова стану процесора для мікропроцесора КР580ВМ80А.

Лабораторна робота № 3

Уведення-виведення даних, організація умовних переходів

Мета роботи. Ознайомлення з методами обміну інформацією з найпростішими пристроями уведення-виведення. Вивчення програмних способів маскування даних і організації умовних переходів в мікро-ЕОМ.

Короткі відомості із теорії

Зв'язок мікропроцесора з навколишнім світом здійснюється через пристрої уведення-виведення (ПрУВ). Взаємодія між ПрУВ і МП реалізується за допомогою портів уведення-виведення. Напрям передачі інформації розглядається відносно мікропроцесора. Тому портом уведення називають любе джерело даних, яке передає інформацію в МП. Портом виведення називають пристрій, який сприймає інформацію від мікропроцесора, коли останній звертається до нього. Порти уведення-виведення мають свої адреси, тому до одного МП можна приєднувати декілька ПрУВ.

В мікропроцесорі типу КР580ВМ80А для адресації портів (тобто для вибирання потрібного пристрою) використовується частина адресної шини (вісім молодших біт). МП дозволяє приєднувати до 256 портів виведення і до 256 портів уведення. Порти "активізуються" (можуть сприймати або видавати інформацію), якщо на адресній шині появляється код, який відповідає номеру порта. При цьому по шині керування для порту уведення повинен бути поданий сигнал читання із порта I/OR, а для порту виведення - сигнал запису в порт I/OW.

Мікропроцесор КР580ВМ80А використовує команди IN та OUT для обміну інформацією з пристроями уведення-виведення. При виконані команди IN <A1> (код команди 0DBH) МП читає число з вхідного пристрою, адреса якого (A1), і записує його в акумулятор. При виконані команди OUT<A1> (код команди 0D3H) мікропроцесор записує число із акумулятора в порт виведення, адреса якого (A1).

Портом уведення або виведення може бути 8-розрядний регістр, який приєднаний до шини даних МП. Для цієї мети часто використовується багаторежимний буферний регістр (ББР), який виготовляється у вигляді мікросхеми типу К589ИР12 (аналог I8212). Умовне графічне зображення ББР приведене на мал. 3.1. Ця мікросхема містить вісьмирозрядний регістр на основі D-тригерів, логічну схему керування та тригер запиту

на переривання. Режим роботи ББР визначається сигналом, який поступає на вхід MD: MD=0 відповідає запису інформації в регістр, а MD=1 - виведеню інформації.

Таблиця 1.1. Призначення виводів мікросхеми К589ИР12

Номер виводуПозначенняТип виводуФункціональне призначення1, 13 2 3, 5, 7, 9 16, 18, 20, 22 4, 6, 8, 10, 15, 17, 19, 21 11 12 14 23 24CS1, CS2 MD D0 - D7 Q0 - Q7 EW GND R INR UccВходи Вхід Входи Виходи (Z-стан) Вхід Вхід Вихід Вибір кристалу Вибір режиму Інформація Інформація Сигнал стробування Спільний Встановлення нуля Запит на переривання Напруга живлення 5 В

Уведення інформації із зовнішнього пристрою (ЗПр) через ББР у мікропроцесор (мал. 3.2) при MD=0 здійснюється наступним чином: зовнішній пристрій, який підготовив байт інформації для передачі у мікропроцесор, виробляє сигнал запису даних EW=1. Коли сигнал на вході EW змінюється з 1 на 0 інформація записується у ББР. Виходи Q0 - Q7 знаходяться при цьому у високоомному стані. Коли мікропроцесор виконує команду читати інформацію із порту, то по шині керування на вхід CS2 поступає сигнал I/OR високого рівня (лог. 1), а на вхід CS1 - сигнал вибору кристалу низького рівня від дешифратора адреси. Тоді виходи Q0 - Q7 відкриваються і інформація із ББР поступає на шину даних мікропроцесора.

Виведення інформації із МП через ШД у ЗПр здійснюється при MD=1 і EW=1. Коли мікропроцесор виконує команду записувати інформацію в порт, то по шині керування на вхід CS2 поступає сигнал I/OW високого рівня, а на вхід CS1 - сигнал вибору кристалу низького рівня від дешифратора адреси. По перепаду сигналу на вході CS2 з 1 в 0 інформація записується у ББР і через виходи Q0 - Q7 поступає у зовнішній пристрій. Приклад реалізації портів уведення і виведення з використанням регістру типу К589ИР12 показаний на мал. 3.2.

Часто для реалізації обміну інформацією між мікропроцесором і зовнішніми пристроями паралельним кодом (при кожній операції передається 8 біт інформації) використовується мікросхема КР580ВВ55 - універсальний паралельний інтерфейс з програмним керуванням режимами роботи. Ця мікросхема має три 8-розрядні порти уведення-виведення. Схема приєднання пристроїв уведення і виведення до мікро-ЕОМ, які використовуються в даній роботі, показана на мал. 3.3. Пристрій уведення представляє собою три кнопки S1-S3. Якщо кнопка ненатиснута, то на вхідний порт по відповідній лінії подається "0", а при натиснені на кнопку - "1". Вихідний пристрій складається із світлових діодів HL1 - HL8, які відображають інформацію, яка записується у порт виведення. Свічення діода відповідає наявності лог. 1 у даному розряді порту виведення.

Найпростіша програма (програма 3.1) уведення коду із вхідного пристрою (з адресою 91H) і виведення його у вихідний пристрій (з адресою 90H) має вигляд:

Програма 3.1.

Адреса	Машинний код	Мітка	Мнемокод	Коментар
0800	3E 82		MVI A, 82H	; встановити режим портів:
0802	D3 93		OUT 93H	; порт А - виведення, ; порт В - уведення.
0804	DB 91	CNT:	IN 91H	; записати число із вхідного ; пристрою з адресою 91H ; в акумулятор
0806	D3 90		OUT 90H	; записати число із акумуля- ; тора у вихідний пристрій ; з адресою 90H
0808	C3 04 08		JMP CNT	; перейти на мітку CNT

В цій і послідуючих програмах перші дві команди: завантажити в акумулятор код слова керування універсальним паралельним інтерфейсом КР580ВВ55 (MVI A, 82H) і вивести вміст регістру А в регістр слова керування з адресою 93H (OUT 93H) служать для програмування режиму роботи пристрою уведення-виведення.

Організація умовних переходів в мікро-ЕОМ здійснюється з використанням регістру ознак МП.

Регістр ознак має п’ять розрядів, кожний з яких приймає значення "0", або "1" по певному правилу у відповідності з виконанням МП останньої команди. Цими розрядами являються.

1. Розряд перенесення С - CARRY. В нього записується 1, якщо при виконанні арифметичної операції було переповнення акумулятора (перенесення із старшого розряду, або запозичення в старший розряд), інакше в розряд записується 0.

2. Розряд знаку S - SIGN. В нього записується 1, якщо при виконанні арифметичних або логічних операцій в старшому, сьомому, розряді акумулятора записана 1, інакше в розряд записується 0.

3. Розряд нульового результату Z - ZERO. В нього записується 1, якщо при виконанні арифметичних або логічних операцій всі розряди числа в акумуляторі мають 0, інакше в розряд записується 0.

4. Додатковий розряд перенесення AC - AUX.CARRY. В нього записується 1, якщо при виконанні операцій в акумуляторі мало місце перенесення 1 із третього розряду в четвертий.

5. Розряд парності P - PARITY. В нього записується 1, якщо при виконанні операції в розрядах акумулятора буде парне число одиниць.

В багатьох випадках при виконанні програм необхідно перевіряти, або змінювати стан одного або декількох розрядів числа в акумуляторі. Це можна реалізувати за допомогою операцій:

- логічного множення числа в акумуляторі і певного числа, яке називають маскою, після чого розряд числа в акумуляторі містить 0, якщо у відповідному розряді маски був записаний 0, і не змінить його, якщо в розряді числа маски записана 1.

- логічного додавання числа в акумуляторі і маски, в результаті в розряд числа в акумуляторі буде записана 1, якщо в такому ж розряді маски є 1, і не змінить його, якщо у відповідному розряді маски записаний 0.

- логічною операцією "АБО з виключенНям" числа в акумуляторі і маски. Результат операції - інверсія вмісту розряду числа, якщо у відповідному розряді маски записана 1, і розряд числа не змінюється, якщо в цьому ж розряді маски записаний 0.

Логічні операції можна виконувати також з вмістом акумулятора і внутрішніх регістрів мікропроцесора. В цьому випадку команди однобайтні. При виконанні всіх логічних команд змінюються розряди Z, S, P, AC регістра ознак, а в розряд C записується 0. Програма маскування окремих розрядів числа, уведеного із вхідного пристрою, приведена нижче (програма 3.2). Програма виводить результат маскування у вихідний пристрій.

Програма 3.2

Адреса	Машинний код	Мітка	Мнемокод	Коментар
0800	3E 82		MVI A, 82H	; встановити режим
0802	D3 93		OUT 93H	; портів
0804	DB 91	CNT:	IN 91H	; одержати число із ; вхідного пристрою
0806	E6 04		ANI 04	; виконати логічну операцію
0808	D3 90		OUT 90H	; записати результат ; у вихідний пристрій
080A	C3 04 08		JMP CNT	; іти на початок

Умовні переходи організуються в програмах при допомозі команд умовних переходів. При виконанні цих команд МП перевіряє стан відповідного розряду регістра ознак. Якщо при перевірці умова не виконується, то виконується наступна по порядку команда програми. Всі команди умовних переходів - трьохбайтні. Перший байт містить код команди, другий та третій байти - адресу передачі керування. Таким чином, команди умовних переходів дозволяють створювати розгалужені програми і в залежності від вмісту регістру ознак переходити на різні ділянки програми.

Програма 3.3

Адреса	Машинний код	Мітка	Мнемокод	Коментар
0800	3E 82		MVI A, 82H	; встановити режим
0802	D3 93		OUT 93H	; портів
0804	DB 91	WAIT:	IN 91H	; одержати число із вхідного ; пристрою
0806	E6 02		ANI 02	; перевірити чи натиснута ; кнопка S2
0808	CA 04 08		JZ WAIT	; іти на WAIT, якщо кнопка не ; натиснута
080B	D3 90		OUT 90H	; записати результат ; у вихідний пристрій
080D	CF		RST 1	; закінчити виконання ; програми

Вище приведена програма (програма 3.3) для перевірки наявності 1 в другому розряді числа, записаного у вхідний пристрій. Програма використовує маскування числа і умовний перехід.

В приведених вище програмах має місце лише один цикл, в якому працювала мікро-ЕОМ. Програма, в якій перевіряється наявність 1 у першому, а потім у другому розрядах числа, уведеного із вхідного пристрою (програма 3.4), містить два цикли.

Програма 3.4

Адреса	Машинний код	Мітка	Мнемокод	Коментар
0800	3E 82		MVI A, 82H	; встановити режим
0802	D3 93		OUT 93H	; портів
0804	DB 91	WAIT1:	IN 91H	; одержати число із ; вхідного пристрою

```
0806    E6 01              ANI 01        ; кнопка S1 натиснута?
0808    CA 04 08           JZ WAIT1      ; якщо ні, то іти на WAIT1
080B    3E FF              MVI A, 0FFH   ; якщо так, то засвітити
080D    D3 90              OUT 90H       ; світлові діоди вихідного
                                         ; пристрою
080F    DB 91      WAIT2:  IN 91H        ; одержати число із вхідного
                                         ; пристрою
0811    E6 04               ANI 04       ; кнопка S3 натиснута?
0813    CA 0F 08           JZ WAIT2      ; якщо ні, то іти на мітку
                                         ; WAIT2
0816    3E 00              MVI A, 00     ; якщо так, то погасити
                                         ; світлові діоди
0818    D3 90              OUT 90H       ; вихідного пристрою
081A    C3 04 08           JMP WAIT1     ; повторити програму
```

Завдання для домашньої підготовки

1. Вивчіть способи організації обміну інформацією між мікро-ЕОМ і зовнішніми пристроями. Розгляньте схеми приєднання пристроїв уведення-виведення даних, які використовуються при проведені лабораторної роботи.

2. Познайомтесь з командами уведення-виведення МП КР580ВМ80А, а також з часовими діаграми їх виконання.

3. Вивчіть групи логічних команд та команд умовних переходів.

4. Ознайомтесь з розрядами регістру ознак МП та правилами запису в них 1.

5. Ознайомтесь з програмами 3.1 - 3.4.

6. Самостійно розробіть програми: а) свічення світлових діодів вихідного пристрою, якщо одночасно натиснуті кнопки S2 і S3; б) свічення світлових діодів вихідного пристрою, які відповідають парним номерам, якщо натиснута кнопка S1 і свічення діодів, які відповідають непарним номерам, якщо натиснута кнопка S2.

Завдання до лабораторної роботи

Завдання 1. Дослідити програму 3.1. Порядок виконання завдання:

- увести в мікро-ЕОМ програму 3.1;

- здійснити пуск програми. Переконатися, що при виконанні програми мікро-ЕОМ постійно переписує дані з вхідного пристрою у вихідний. Для цього з допомогою перемикачів S1-S3 змінювати коди, які записуються у порт В. Інформація, яка

виводиться у вихідний пристрій, відображається світловими діодами HL1-HL8 (див. мал. 3.2).

Завдання 2. Дослідити програму 3.2. Порядок виконання завдання:

- увести програму в мікро-ЕОМ, здійснити її пуск і дослідити результат її виконання по числу, яке відображається вихідним пристроєм;

- помінявши в програмі 3.2. двобайтну команду ANI< D > на однобайтні ANA A, XRA A, ORA A дослідити результат їх виконання по числу, яке записується у вихідний пристрій.

Завдання 3. Дослідити програму 3.3. Порядок виконання завдання:

- увести програму, здійснити її пуск;

- переконатися, що мікро-ЕОМ реагує лише на ті числа, які містять 1 в першому розряді. Після закінчення виконання програми перевірте, що в розряді Z регістра ознак записана 1.

Завдання 4. Дослідити програму 3.4. Порядок виконання завдання:

- увести програму в мікро-ЕОМ, здійснити її пуск;

- переконатися, що при наявності 1 в нульовому розряді числа, яке уводиться з вхідного пристрою, світяться діоди вихідного пристрою;

- записати 1 у другий розряд числа, яке уводиться з вхідного пристрою (натиснути кнопку S3), і переконатися, що світлові діоди вихідного пристрою виключаються.

Завдання 5. Дослідити програми, розроблені в п. 6 домашньої підготовки.

Зміст звіту

Звіт повинен містити:

- назву та мету роботи;

- схеми приєднання зовнішніх пристроїв до мікро-ЕОМ;

- самостійно розроблені і досліджені в процесі виконання лабораторної роботи програми, вказані в п. 6 “Завдання для домашньої підготовки”;

- повний перелік команд передачі керування по умові для МП КР580ВМ80А;

- повний перелік команд логічних операцій для МП КР580ВМ80А.

Додаткові завдання

Д3.1. Напишіть програму, яка перетворює двійковий код числа, уведеного з вхідного пристрою, в одиничний позиційний код.

Д3.2. Зобразіть часові діаграми процесу виконання мікропроцесором команд: IN PORT, OUT PORT, де PORT адреса порта уведення, або виведення.

Д3.3. Розробіть схеми приєднання дешифратора адреси та зовнішніх пристроїв при організації обміну інформацією з зовнішніми пристроями з допомогою команд читання і запису у пам'ять.

Д3.4. Розробіть схему дешифратора адреси та схему одержання сигналів I/OW, I/OR для обміну інформацією між ЕОМ і зовнішніми пристроями з допомогою мікросхеми К589ИР12 (адреса порта уведення 20Н, порта виведення - 30Н).

Д3.5. Розробіть схему уведення даних в мікропроцесор по перериванню на основі регістру К589ИР12.

Д3.6. Напишіть програму "Бігаючі вогники" для вихідного пристрою, який зображений на мал. 3.2.

Д3.7. Користуючись описом програми Монітор мікро-ЕОМ УМК напишіть програму виведення числа, одержаного із вхідного пристрою, на дисплей мікро-ЕОМ.

Контрольні запитання

1. З допомогою яких команд мікро-ЕОМ може здійснювати уведення-виведення інформації?

2. За скільки машинних тактів здійснюється уведення-виведення інформації з допомогою команд IN < A >, OUT < A >.

3. Які режими адресації операндів Ви знаєте?

4. При виконані яких команд, приведених в програмі 3.2, змінюються розряди регістру ознак?

5. За яких умов записуються 1 в розряди регістру ознак?

6. Що називають портом уведення? Портом виведення?

7. Назвіть види логічних операцій, які виконуються МП КР580ВМ80А?

8. Яке число називається маскою?

9. Яке число буде в акумуляторі після виконання команди ANI 01 в програмі 3.4?

10. З якою метою і як реалізуються переривання в МП типу КР580ВМ80А?

Лабораторна робота № 4

Підпрограма і стек

Мета роботи. Вивчення способів запису та звертання до підпрограм; вивчення методів використання стеку при написанні програм.

Короткі відомості із теорії

При написані програм стараються їх робити як можна коротшими. З цією метою частина програми, яка багато разів повторюється, або програма, яка часто використовується, можуть бути оформлені у вигляді підпрограм - послідовності команд, виконання яких може бути викликане із любого місця програми довільне число раз. Процес передачі керування підпрограмі називається її викликом. Дані і адреси, необхідні для роботи підпрограми, називають вхідними параметрами. Результати роботи підпрограми, які передаються після її роботи в основну програму, називають вихідними параметрами.

Для виклику підпрограми і повернення із неї використовуються команди CALL<A2><A1> і RET. Команда CALL <A2><A1> завантажує в лічильник команд МП байти <A2><A1>, які записані по двох наступних адресах пам'яті після адреси по якій записаний код команди CALL (CD). Причому байт <A2> записується в молодший байт лічильника команд, а третій байт команди <A1> - в старший байт лічильника команд. Одночасно мікропроцесор записує в стек адресу наступної, після команди CALL, команди основної програми, до якої він буде звертатись після виконання підпрограми.

Стек - спеціально організована область пам'яті ОЗПр в мікро-ЕОМ для тимчасового зберігання даних або адрес. Число, записане в стек останнім, читається із нього першим.

Команда повернення із підпрограми RET (C9) заносить в лічильник команд два байти, які записані в стеку в даний момент. Після цього виконання програми буде продовжене з цієї адреси. Люба підпрограма повинна закінчуватися командою RET.

Автоматичне зберігання та відновлення адреси основної програми при виконанні підпрограм дозволяє робити підпрограми, які вкладені одна в одну, тобто викликати одну підпрограму із другої. Число вкладених одна в одну підпрограм для даної мікро-ЕОМ визначається лише розміром стеку.

Існують також команди, які дозволяють викликати підпрограми і повертатись із них по певному стану розрядів регістру ознак. Всі команди умовного виклику підпрограм - трьохбайтні, другий і третій байти містять адресу підпрограми.

Крім команд переходу на підпрограми і повернення із них в стек можна заносити інформацію з допомогою команд PUSH <R> (записати в стек вміст регістру R) і POP<R> (занести дані із стеку в регістр R). Ці команди однобайтні. При написанні програм необхідно на початку програми визначити, яка область пам'яті відводиться під стек, записавши в регістр SP адресу вершини стеку за допомогою команди LXI SP, <A2><A1>, або командою SPHL (поміняти вміст регістрів SP і HL).

Всі операції зі стеком повинні бути збалансовані - люба підпрограма повинна містити однакову кількість команд PUSH <R> і POP <R> і закінчуватись командою RET.

Алгоритм роботи простої підпрограми часової затримки приведений на мал. 4.2. Загальний час затримки обчислюється по формулі

$$TD=t1+(t2+t3+t4)N1+t5,$$

де N1 - число, записане в лічильник. В цій підпрограмі лічильником є регістр B. Команда NOP потрібна для збільшення часу виконання циклу і, відповідно, всієї затримки. Замість команди NOP може бути записна люба послідовність команд, виконання яких не змінить вміст регістрів МП.

Підпрограма DLY (підпрограма 4.1) є підпрограмою часової затримки, яка записана у відповідності з алгоритмом мал. 4.2.

Програма 4.1

Адреса	Машинний код	Мітка	Мнемокод	Коментар
0900	06 N1		MVI B, N1	; Завантажити в регістр B ;число N1
0902	00	DLY:	NOP	; немає операції
0903	05		DCR B	; зменшити число в ; регістрі B на 1
0904	C2 02 09		JNZ DLY	; якщо вміст B не рівний нулю, ; то іти, на DLY
0907	C9		RET	; повернутись в основну ; програму

Максимальний час затримки має місце, коли N1=00, мінімальний - при N1=01.

Програма 4.2

Адреса	Машинний код	Мітка	Мнемокод	Коментар
0900	F5	DEL:	PUSH PSW	; зберегти в стеку вмісти
0901	E5		PUSH H	; регістрів A, F, H, L, D, E
0902	D5		PUSH D	
0903	2A 50 0B		LHLD 0B50H	; завантажити пару HL
				; числом з адреси 0B50
0906	EB		XCHG	; поміняти вміст HL і DE
0907	1B	M1:	DCX D	; зменшити вміст пари
				; DE на 1
0908	7A		MOV A, D	; передати дані з
				; регістру D в A
0909	B3		ORA E	; логічне додавання вмісту
				; регістрів A і E
090A	C2 07 09		JNZ M1	; якщо число в регістрі DE
				; не рівне нулю,
				; то іти на M1
090D	D1		POP D	; відновити вміст регістрів
090E	E1		POP H	; D, C, H, L, A, F
090F	F1		POP PSW	;
0910	C9		RET	; повернутися в основну
				; програму

Підпрограма DEL (програма 4.2) є підпрограмою часової затримки на час до 0,7 секунди. При виконанні програми зберігається вміст регістрів A, D, E, H, L і F. Параметр програми (число, яке визначає час затримки) заноситься командами LHLD<A2><A1> і XCHG з комірок пам'яті з адресою 0B50H (молодший байт) і 0B51H (старший байт).

Використання підпрограм розглянемо на прикладі програми 4.3 (програма "Символ"), алгоритм якої зображений на мал. 4.3. Ця програма контролює стан кнопки S1 і, якщо вона натиснута, формує на індикаторі зображення символу, коди якого знаходяться в пам'яті починаючи з адреси 0848H. Погасити індикатор можна натиснувши на кнопку S2 (див. мал. 4.1).

Спосіб формування зображення на світлодіодній матриці пояснює мал. 4.4. Матриця має 7 рядків і 5 стовпців. Світлові діоди в рядку мають спільні катоди, а в стовпці - спільні аноди. Світитися буде той діод (або група діодів) катод якого має напругу близьку до нуля, а на відповідний стовпець подана висока напруга. Виводи рядків через буферні логічні елементи приєднані до ліній порту B, а виводи стовпців - до ліній порту A. Наприклад, щоб світились чотири верхніх діоди у першому стовпці (відлік

ведеться зверху вниз і зліва на право) потрібно у порт А вивести число 00010000B (10H), а у порт В - число 00001111B (0FH).

Програма 4.3 (програма “Символ”)

Адреса	Машинний код	Мітка	Мнемокод	Коментар
0800	31 DA 0B		LXI SP, 0BDAH	
0803	3E 89		MVI A, 89H	
0805	D3 93		OUT 93H	
0807	21 00 01		LXI H, 0100H	
080A	22 50 0B		SHLD 0B50H	
080D	CD 30 08	P1:	CALL STR	
0810	CD 35 08		CALL KNP1	
0813	21 48 08	BUK:	LXI H, TAB	
0816	06 01		MVI B,1	
0818	0E 05		MVI C, 5	
081A	CD 3E 08	ROT:	CALL SWIT	
081D	78		MOV A, B	
081E	07		RLC	
081F	47		MOV B, A	
0820	23		INX H	
0821	0D		DCR C	
0822	C2 1A 08		JNZ ROT	
0825	DB 92		IN 92H	
0827	2F		CMA	
0828	E6 02		ANI 02	
082A	CA 13 08		JZ BUK	
082D	C3 0D 08		JMP P1	
0830	3E 00	STR:	MVI A, 0	
0832	D3 90		OUT 90H	
0834	C9		RET	
0835	DB 92	KNP1:	IN 92H	
0837	2F		CMA	
0838	E6 01		ANI 1	
083A	CA 35 08		JZ KNP1	
083D	C9		RET	
083E	7E	SWIT:	MOV A, M	
083F	D3 91		OUT 91H	
0841	78		MOV A, B	
0842	D3 90		OUT 90H	
0844	CD 00 09		CALL DEL	
0847	C9		RET	
0848	1F 30	TAB:	DB 1FH, 30H	
084A	50 50		DB 50H, 50H	
084C	4F		DB 4FH	

Програма “Символ” розпочинається групою команд, які встановлюють адресу вершини стеку, програмують режим роботи універсального паралельного інтерфейсу КР580ВВ55, завантажують константу підпрограми часової затримки DEL і працює з трьома підпрограмами: STR, KNP1, SWIT. Підпрограма STR розпочинається з адреси 0830H і використовується, коли потрібно припинити свічення індикатора. Підпрограма SWIT працює з регістрами B, C, H, L. В регістрі В знаходиться адреса стовпця, регістр С служить лічильником, а пара HL використовується для вибору чисел з пам’яті (починаючи з адреси 0848H) і виведення їх у порт В. У порт А виводиться адреса (номер) стовпця, з регістру В. Підпрограма KNP1 контролює стан кнопки S1. Якщо вона натиснута, то передає керування основній програмі.

Завдання для домашньої підготовки

1. Ознайомтесь з командами виклику і повернення із підпрограм по умові для мікропроцесора КР580ВМ80А.

2. Визначте, при яких числах, записаних у регістр В, підпрограма 4.1 буде реалізовувати мінімальний і максимальний час затримки. Підрахуйте цей час, якщо тривалість машинного такту МП Т=450 нс.

3. На основі підпрограм 4.1 і 4.2 напишіть програму, яка засвічує світловий діод (любий) вихідного пристрою (див. мал. 4.1) на час 5 с.

4. Вивчіть програму 4.3, яка виводить на індикатор зображення літери "У". Зверніть увагу, що при натисненні на кнопки S1-S3 у вхідний порт С по відповідних лініях С0-С2 записується лог.0. Проаналізуйте, як буде працювати програма 4.3, коли в комірку пам’яті з адресою 0809H записати число 0AH?

Завдання до лабораторної роботи

Завдання 1. Дослідити процеси виконання команд виклику та повернення із підпрограми, а також команд роботи із стеком. Порядок виконання завдання:

- уведіть в мікро-ЕОМ програми 4.2 і 4.4;
- запишіть в комірки пам’яті з адресами 0B50H і 0B51H числа 02 і 00, відповідно;
- виконати програму 4.4 по машинних циклах. Перевірити вмісти всіх регістрів МП;

- замінити в підпрограмі DEL (програма 4.2) команду POP PSW на команду NOP і прослідкувати, як буде виконуватися програма 4.4. Пояснити зміни, які сталися.

Програма 4.4

```
LXI SP, 0BDAH
CALL DEL
RST1
```

Завдання 2. Дослідити програму часової затримки на прикладі програми, яка послідовно засвічує і гасить світлові діоди вихідного пристрою. Порядок виконання завдання:

- увести в мікро-ЕОМ програму, яка розроблена в п. 3 завдання для домашньої підготовки;

- здійснити її пуск та перевірити, що вона працює;

- змінити числа, які записуються в регістри D, E в підпрограмі DEL. Перевірити можливість зміни тривалості затримки.

Завдання 3. Дослідити програму 4.3. Порядок виконання завдання:

- увести програму, здійснити її пуск;

- переконатись, що мікро-ЕОМ контролює стан кнопки S1 (див. мал. 4.1) і виводить на індикатор зображення літери "У", якщо натиснути кнопку;

- модифікувати програму 4.3 так, щоб мікро-ЕОМ контролювала кнопки S2 і S3 і виводила зображення іншого символу.

Зміст звіту

Звіт повинен містити:

- назву та мету роботи;

- повний перелік команд виклику і повернення із підпрограм для МП типу КР580ВМ80А;

- перелік команд роботи зі стеком для цього ж мікропроцесора;

- результати розрахунків часу затримки по п. 2 завдання для домашньої підготовки;

- розроблені програми по п. 3 і 4 завдання для домашньої підготовки;

- результати виконання завдань 1 - 3;

- програму свічення діода тривалістю 5 с;

- модифіковану програму при виконанні завдання 3.

Додаткові завдання

Д4.1. Вивчити часові діаграми виконання команд CALL<A2><A1> і RET мікропроцесора КР580ВМ80А.

Д4.2. Вивчити часові діаграми виконання команд PUSH <R> і POP <R>

Д4.3. Розробіть схему і напишіть програму формування звукових сигналів на основі паралельного інтерфейсу КР580ВВ55.

Д4.4. Розробіть схему генератора часових інтервалів на основі ВІС КР580ВИ53 з тривалістю від 0,5 мкс до 0,5 год.

Д4.5. Напишіть програму виведення двох символів при натисненні на кнопки S1 і S2 (див. мал. 4.1).

Д4.6. Напишіть програму, яка б контролювала стан кнопок S1 - S3 пристрою уведення-виведення (мал. 4.1) і при натисненні на кнопку виводила б зображення числа, яке присвоєне цій кнопці.

Д4.7. Розробіть схему уведення паралельних даних в мікро-ЕОМ по перериванню на базі ВІС КР580ВВ55.

Контрольні запитання

1. В якій послідовності записуються і читаються із стеку вміст акумулятора та регістру ознак МП при виконанні команд PUSH PSW та POP PSW?

2. Якими командами можна задати, або замінити область пам’яті, яка відводиться під стек?

3. Поясніть порядок виконання команди RET мікропроцесором.

4. Порівняйте процеси виконання команд CALL і RST.

5. В якій послідовності зберігаються і відновлюються вмісти регістрів мікропроцесора в підпрограмах?

6. Яка мінімальна адреса буде записана в стеку при виконанні програми 4.4?

7. Який максимальний час затримки може забезпечити програма 4.2, якщо тривалість машинного такту Т=0,45 мкс?

8. З якою метою використовується команда CMA в програмі 4.3?

9. Як змінити підпрограму KNP1 в програмі 4.3, щоб мікро-ЕОМ контролювала стан, коли натисненні дві кнопки одночасно?

10. Який режим роботи мікро-ЕОМ називають режимом прямого доступу до пам’яті (ПДП)?

Лабораторна робота № 5

Організація обміну інформацією. Інтерфейси мікро-ЕОМ

Мета роботи. Вивчити способи обміну інформацією між мікро-ЕОМ і зовнішніми пристроями. Моделювання радіального паралельного інтерфейсу.

Короткі відомості із теорії

Для забезпечення зв'язку мікро-ЕОМ з різними зовнішніми пристроями (ЗПр) використовують два типи інтерфейсів - паралельний і послідовний. Інтерфейс - це сукупність уніфікованих апаратних, програмних та конструктивних засобів, які забезпечують обмін інформацією між мікро-ЕОМ і ЗПр, або між двома мікро-ЕОМ.

В мікропроцесорних пристроях для реалізації паралельного уведення-виведення даних часто використовується ВІС програмовного паралельного інтерфейсу (ППІ) КР580ВВ55. Мікросхема представляє собою програмовний пристрій для уведення-виведення інформації представленої паралельним кодом. Вона дозволяє здійснювати обмін 8-розрядними даними по трьох каналах: А, В, С. Напрям обміну та режим роботи для кожного каналу задаються програмно. Канали служать для передачі як даних, так і сигналів керування.

До складу ППІ входять три 8-розрядних порти (А, В, С), які призначені для організації обміну даними між мікропроцесором і ЗПр, схема керування уведенням-виведенням з регістром слова керування (РСК).

На мал. 5.1 приведено умовне позначення ППІ на електричних схемах. Порти А і В складаються із вихідного і вхідного 8-розрядних регістрів та двонапрямного шинного буферу. Порт С складається із двох вхідних і двох вихідних 4-розрядних регістрів та двох двонапрямних шинних формувачів (буферів).

Призначення виводів мікросхеми. D0 - D7 - двонапрямна магістраль даних для передачі даних, слова керування та інформації про стан мікросхеми. RD - читати. По сигналу низького рівня вміст вхідного регістру одного із портів виводиться на шину даних. WR - записувати. По сигналу низького рівня інформація із шини даних записується у вихідний регістр одного із портів або РСК. А0, А1 - лінії шини адреси. На ці входи подається адреса, по якій здійснюється звертання до одного із трьох портів або

до РСК (таблиця 5.1). CS - вибір мікросхеми. Сигнал низького рівня приєднує виводи D0 -D7 ППІ до шини даних мікропроцесорної системи. RESET - установити. Сигнал високого рівня встановлює у вихідний стан регістр слова керування. Після зняття сигналу RESET виводи всіх портів настроюються на уведення інформації (режим 0). PA0 - PA7 - входи-виходи порту A. PB0 - PB7 - входи-виходи порту B. PC0 - PC7 - входи-виходи порту C.

Таблиця 5.1.

Сигнали на входахПорт, який вибираєтьсяA1A00 0 1 10 1 0 1A B C РСК

Настроювання ППІ здійснюється програмно шляхом запису слова керування командою OUT із акумулятора МП в регістр слова керування.

Група А ППІ (порт А і половина порту С) можуть настроюватися на один із трьох режимів: 0 - основний режим уведення-виведення; 1 - режим стробованного уведення-виведення; 2 - режим двонапрямної шини.

Група В (порт В і половина порту С) може настроюватися на режими 0 або 1.

Структура слова керування зображена на мал. 5.2. В залежності від інформації в старшому розряді слова керування ППІ програмується на потрібний режим (D7=1) (мал. 5.2а), або встановлюються розряди порту С (D7=0) (мал. 5.2б). Любий з восьми бітів порту С може бути встановлений в "1", або в "0" по команді OUT.

Режим 0. В цьому режимі по сигналу, який подається на вхід RD (WR) дані читаються (записуються) із зовнішнього (у зовнішній) пристрій через порт, який вибирається з допомогою сигналів на входах А0, А1. При виведені інформація запам’ятовується в регістрах портів, а при уведені - сигнали на входах порту потрібно підтримувати до того часу, доки мікропроцесор не прочитає її.

Режим 1. В цьому режимі здійснюється асинхронний обмін інформацією між портами і зовнішніми пристроями. При цьому порти А і В використовуються для передачі даних, а порт С - для передачі сигналів керування. Функціональне призначення розрядів порту С в режимі уведення інформації наступне (мал. 5.3 a):

- вхід STR використо-вується для приймання сигналу, який стробує запис інформації в порт;

- сигнал на виході IBF повідомляє джерело інфор-мації про те, що вхідний буфер заповнений;

- сигнал на виході INTR використовується як запит на переривання для мікропроцесора.

Для реалізації цього режиму тригери розрядів PC4 (для порту A) і PC2 (для порту B) повинні бути попередньо встановлені в "1" командою OUT, якщо програміст дозволяє переривання. Наприклад:

для порту A	для порту B
MVI A, 01011001B	MVI A, 01010101B
OUT RSK	OUT RSK.

Запис інформації із мікропроцесора в порт відбувається по сигналу I/OW. Сигнал низького рівня на виході OBF служить стробом запису інформації у зовнішній пристрій. Після того, коли зовнішній пристрій записав інформацію він посилає сигнал підтвердження низького рівня на вхід АСК (мал. 5.3 *б*).

Режим 2. В режимі 2 порт А настроюється на конфігурацію двонапрямного стробованного обміну. Цей режим використовується для керування швидкодіючими ЗПр по перериванню, але з передачею даних по одній шині.

Для приєднання до мікро-ЕОМ стандартного периферійного обладнання (алфавітно-цифрових терміналів, принтерів і ін.) використовується інтерфейс радіальний паралельний - ІРПР. Передача даних в ІРПР здійснюється між передавальним пристроєм, який називають джерелом, і приймальним пристроєм - приймачем. В ролі джерела інформації може виступати ЕОМ, тоді приймачем буде ЗПр, або друга ЕОМ.

На мал. 5.4 *а* показаний набір інформаційних і керуючих сигналів інтерфейсу ІРПР. Сигнали готовності джерела ГД і готовності приймача ГП встановлюються на весь час роботи і інформують про готовність до передачі (прийому) даних. Сигнали запиту приймача ЗП та стробу джерела СТР використовуються в процесі передачі даних. Самі ж дані передаються по лініях даних D, число яких може досягати 16, але частіше використовується 8 ліній. Для передачі додаткової інформації про стан пристроїв, які обмінюються інформацією, можуть використовуватись сигнали стану джерела СД1 ... 8 та приймача СП1 ... 8.

Протокол ІРПР встановлює в загальному вигляді правила взаємодії сигналів джерела і приймача (мал. 5.4 *б*). Ініціатором передачі є приймач. Сигналом ЗП він інформує джерело про готовність записати порцію даних. Джерело, підготувавши дані, виводить їх на лінії D і повідомляє про це приймач сигналом СТР. Приймач, одержавши сигнал СТР, читає дані з ліній D і знімає сигнал ЗП. У відповідь на це джерело знімає

сигнал СТР і дані. Після цього може розпочинатися наступний цикл передачі. Швидкість обміну може задаватися як приймачем так і джерелом.

Розглянуті вище способи паралельної передачі даних забезпечують високі швидкості обміну інформацією, проте потребують багато провідників для організації шин зв'язку. Тому інтерфейси з паралельною передачею даних використовуються лише в тих випадках, коли ЗПр розміщені недалеко від ЕОМ (до 15 метрів). В тих випадках, коли джерело і приймач інформації розміщені на значній відстані, використовується послідовна передача, при якій дані передаються по лінії зв'язку послідовно біт за бітом. Цей спосіб дозволяє значно зменшити вартість ліній зв'язку, проте значно зменшується швидкість передачі даних.

Розрізняють два способи послідовної передачі даних: синхронний та асинхронний. У першому випадку дані передаються неперервно, без розділення на слова або байти. Для ідентифікації окремих бітів джерело і приймач повинні працювати синхронно під керуванням одного генератора. В другому випадку початок і кінець кожної порції інформації (кадру) відмічаються спеціальними мітками; вимоги до синхронізації передавача і приймача значно зменшуються. У вимірювальній техніці частіше використовується асинхронна передача даних.

Стандартний формат послідовної асинхронної передачі даних зображений на мал. 5.5. Рівень логічної 1 в лінії називають маркером, рівень логічного нуля - пропуском. Якщо дані не передаються в лінії діє сигнал маркера. Передача порції даних розпочинається стартовим бітом, після чого передаються біти даних, число яких в кадрі може встановлюватися від 5 до 8. За бітами даних може слідувати біт контролю парності. Цей біт вибирається в кожній порції даних таким чином, щоб загальне число одиниць в бітах даних і в біті парності було парним (або непарним). Закінчується кадр стоп-бітом, який має рівень маркера. Після цього в лінії може підтримуватися стан відсутності даних (рівень маркера), або розпочатися наступний кадр (стартовим бітом пропуску).

Сигнали в лінії можуть мати різне представлення. При передачі на малі відстані рівню маркера відповідають напруги 3 ... 5 В. При великих відстанях (до 1,5 км) використовують петлю струму - імпульси постійного струму значенням 20 або 40 мА, які передаються по двопроводові лінії (скрученій парі) або по кабелю. У випадку передачі інформації як у прямому, так і в оберненому напрямку (дуплексний зв'язок) використовують чотирипровідникову лінію. Асинхронний зв'язок постійним струмом

(петля струму) по чотирипроводові дуплексні лінії носить назву радіального послідовного інтерфейсу (ІРПС).

При передачі інформації по телефонних лініях рівні напруги перетворюються в синусоїдальні сигнали. Сигналу маркера відповідає частота 1270 Гц, сигналу пропуску - 1070 Гц. Для дуплексного зв'язку по телефонній лінії використовується дві пари частот, наприклад: 1270, 1070 Гц і 2225, 2025 Гц.

Розглянемо приклад використання стандарту ІРПР для організації зв'язку між двома мікро-ЕОМ. Роль центральної ЕОМ (яка передає інформацію) виконує персональна ЕОМ типу ПК-01 "Львів", периферійною ЕОМ (яка приймає дані) є УМК. Структура комплексу зображена на мал. 5.6. Уведення і виведення даних на лінії ІРПР обидві ЕОМ здійснюють через ППІ типу КР580ВВ55. Дані передаються через порти А. Лінії РВ0, РВ1 порту В використовуються для сигналів керування ГП і ЗП, відповідно. Лінії РС6, РС7 порту С відведені для сигналів ГД і СТР.

Програма обміну інформацією (драйвер), яка розміщена в ПК-01, розпочинається з адреси 0В400Н (програма 5.1). На початку програми задаються границі масиву, з якого будуть передаватися дані (адреса початку масиву 7000Н, кінця - 70FFH), встановлюється режим ППІ (режим 0): порти А і С програмуються на виведення, а порт В - на уведення і виводиться сигнал готовності джерела (ГД=1). В циклах з мітками GP і ZP мікро-ЕОМ контролює стан ліній ГП і ЗП. Якщо ГП=1 і ЗП=1 мікро-ЕОМ виводить дані і встановлює СТР=1. Після цього знову контролюється стан лінії ЗП (мітка NZP). Якщо дані записані приймачем (ЗП=0), встановлюється СТР=0 і ЕОМ перевіряє чи передана вся інформація. Сеанс зв'язку закінчується встановленням сигналів ГД=0 та СТР=0.

Програма 5.1

```
;Програма драйвера інтерфейсу ІРПР. Джерело інформації: ЕОМ ПК-01
        ORG 0B400H
        LXI SP, 0BEFFH
        LXI H, 7000H        ; границі
        LXI D, 7100H        ; масиву
        MVI A, 82H          ; режим
        OUT 0C3H            ; портів
        MVI A, 7FH          ; ГД=1
        OUT 0C2H            ; СТР=0
GP:     IN 0C1H
        ANI 1               ; ГП=1 ?
        JZ GP
        DCX H
ZP:     IN 0C1H
```

```
        ANI 2           ; ЗП=1 ?
        JZ ZP
        INX H
        MOV A, M        ; вивести
        OUT 0C0H        ; дані
        MVI A, 0FFH     ; СТР=1
        OUT 0C2H
NZP:    IN 0C1H         ; ЗП=0 ?
        ANI 2
        JNZ NZP         ; так
        MVI A, 7FH      ; СТР=0
        OUT 0C2H
        MOV A, H        ; перевірка
        SUB D
        JNZ ZP          ; границь
        MOV A, L        ; масиву
        SUB E
        JNZ ZP
        MVI A, 3FH      ; ГД=0
        OUT 0C2H        ; СТР=0
        JMP 0F800H      ; кінець обміну
        ;
        ; Приймач інформації: ЕОМ УМК
        ORG 0800H
        LXI SP, 0BDAH
        LXI H, 0900H    ; адреса масиву
        MVI A, 98H      ; режим
        OUT 93H         ; портів
        MVI A, 01       ; ГП=1
        OUT 91H
GD:     IN 92H          ; ГД=1?
        ANI 40H
        JZ GD
M1:     MVI A, 03       ; ЗП=1
        OUT 91H
STR:    IN 92H          ; СТР=1 ?
        ANI 80H
        JZ STR
        IN 90H
        MOV M,A         ; читати дані
        MVI A, 01       ; ЗП=0
        OUT 91H
NST:    IN 92H          ; СТР=0 ?
        ANI 80H
        JNZ NST
        INX H
P1:     IN 92H          ; ГД=0 ?
        ANI 40H
        JNZ M1          ; ТАК
        MVI A, 0        ; ГП=0
        OUT 91H         ; ЗП=0
```

```
RST 1                ; кінець обміну
END
```

Аналогічно працює програма уведення інформації, яка розміщується в УМК починаючи з адреси 0800H. Для приймача інформації достатньо задати лише адресу початку масиву, де будуть розміщуватися дані. Після встановлення режиму ППІ і сигналу ГП=1 в циклі з міткою GD контролюється наявність одиниці в шостому розряді порту С. Ця одиниця інтерпретується як сигнал ГД від джерела. Потім виводиться сигнал запиту приймача на лінію ЗП і в циклі з міткою STR контролюється сигнал СТР. Якщо джерело встановить СТР=1, читаються дані, знімається сигнал ЗП і мікро-ЕОМ чекає (цикл з міткою NST) доки сигнал на лінії СТР стане рівний нулеві. Цикл уведення даних повторюється доки джерело на лінії ГД не встановить сигнал ГД=0.

Завдання для домашньої підготовки

1. Ознайомтесь з командами керування мікропроцесора КР580ВМ80А.
2. Вивчіть основні режими роботи ВІС паралельного інтерфейсу типу КР580ВВ55 та способи її програмування.
3. Ознайомтесь з основними типами універсальних інтерфейсів периферійного обладнання мікро-ЕОМ.
4. Вивчіть логічну та функціональну організацію інтерфейсу для радіального приєднання зовнішніх пристроїв до мікро-ЕОМ з паралельною передачею інформації (ІРПР).
5. Вивчіть програму 5.1, яка забезпечує обмін даними між двома мікро-ЕОМ в стандарті ІРПР.

Завдання до лабораторної роботи

Завдання 1. Підготовка програм, які забезпечують обмін інформацією в стандарті ІРПР, для мікро-ЕОМ. Порядок виконання завдання:

- увести в мікро-ЕОМ типу ПК-01 "Львів" Редактор тексту. Ця програма уводиться з допомогою директив програми Монітор. Директива уведення файла з магнітофона запускається кнопкою "F1 ЧМЛ". З ROM-диску Редактор тексту викликається кнопкою "F4 ED";

- набрати програму 5.1 та занести її в пам’ять мікро-ЕОМ, користуючись інструкцією до Редактора тексту (див. додаток 2). Набраний текст зберігається в пам’яті мікро-ЕОМ починаючи з адреси 5000Н;

- вийти з Редактора тексту, натиснувши кнопку "СБР";

- завантажити програму Асемблер за допомогою кнопки "AS". Після завантаження на дисплеї повинно появитись повідомлення:

МАКРОАССЕМБЛЕР *ПК-01* ВЕР.1.0

P=

З програмою Асемблер працювати згідно інструкції (додаток 2);

- занести в мікро-ЕОМ УМК драйвер для приймача інформації. Ця програма після трансляції Асемблером розміщена в пам’яті ПК-01 починаючи з адреси 7800Н. Читається програма в машинних командах директивою Монітора SP7800. В УМК програму розмістити з адреси 0800Н.

Завдання 2. Дослідити програму 5.1. Порядок виконання завдання:

- з’єднати мікро-ЕОМ ПК-01 і УМК кабелем;

- здійснити пуск програм: в ПК-01 директивою Монітора GB400<ВК>; в УМК - директивою СТ0800<ВП>.

- перевірити, що масив даних з ПК-01 з адреси 7000Н по 70FFH переданий в УМК і розміщений з адреси 0900Н.

Завдання 3. На основі програми 5.1 розробити програму, під керуванням якої б інформація з УМК передавалася б в мікро-ЕОМ ПК-01 (програма 5.2). Дослідити програму 5.2.

Зміст звіту

Звіт повинен містити:

- назву та мету роботи;

- перелік команд керування для мікропроцесора КР580ВМ80А;

- схему зв’язку між мікро-ЕОМ та призначення ліній зв’язку;

- структуру слова керування ППІ типу КР580ВВ55;

- лістинг програм 5.1 та 5.2.

Додаткові завдання

Д5.1. Розробіть драйвер інтерфейсу ІРПР при використані ВІС КР580ВВ55 в режимі 1.

Д5.2. Розробіть схему передачі інформації в стандарті ІРПС на базі мікросхеми КР580ВВ51 (універсальний синхронно-асинхронний приймач-передавач).

Д5.3. Напишіть драйвер інтерфейсу ІРПС при використані ВІС КР580ВВ51.

Д5.4. Розробіть схему, яка дозволяє проводити обмін інформацією між мікро-ЕОМ і ЗПр з допомогою контролера прямого доступу до пам'яті типу КР580ВТ57.

Д5.5. Напишіть драйвер інтерфейсу ІРПР-М (Centronics) при використані ВІС КР580ВВ55 в режимі 1.

Контрольні запитання

1. Які способи обміну інформацією між мікро-ЕОМ і ЗПр Ви знаєте?

2. Намалюйте структуру мікросхеми паралельного програмовного інтерфейсу КР580ВВ55.

3. Як здійснюється адресація портів ППІ КР580ВВ55?

4. Напишіть програму для встановлення таких напрямків передачі даних через ППІ в режимі 0: порт А - уведення; порт В - виведення; порт С (С0-С3) - виведення; порт С (С4-С7) - уведення.

5. Напишіть програму для встановлення 5-го розряду порту С в "1".

6. Поясніть роботу мікросхеми КР580ВВ55 в режимі 0.

7. Поясніть роботу мікросхеми КР580ВВ55 в режимі 1.

8. Розкажіть логічну і функціональну організацію інтерфейсу ІРПР.

9. Поясніть способи послідовної передачі даних: синхронний та асинхронний.

10. Поясніть стандартний формат послідовної асинхронної передачі даних.

11. В якій частині програми 5.1 мікро-ЕОМ контролює готовність приймача?

12. В якій частині програми 5.1 мікро-ЕОМ контролює сигнал ЗП?

13. В якій частині програми 5.1 мікро-ЕОМ контролює сигнал СТР?

14. В якій частині програми 5.1 мікро-ЕОМ контролює сигнал ГД?

15. В якій частині програми 5.1 мікро-ЕОМ читає дані? Виводить дані?

Лабораторна робота № 6

Моделювання вимірювальної системи на базі мікро-ЕОМ

Мета роботи. Моделювання простої вимірювально-обчислювальної системи та вивчення на її основі основних рис програмного забезпечення фізичного експерименту: модульність, використання різних мов програмування, способів керування процесом вимірювання.

Короткі відомості із теорії

Використання мікро-ЕОМ у вимірювальних системах дозволяє підвищити точність вимірювань, значно збільшити їх продуктивність, автоматизувати процеси одержання, зберігання та обробки інформації. Не дивлячись на те, що конкретні вимірювальні установки відрізняються одна від одної, їх структури часто виявляються однаковими.
Фізична величина, яку потрібно виміряти, перетворюється відповідними датчиками (первинними перетворювачами) в електричний сигнал. Інформація про вимірювану величину може міститись в різних параметрах сигналу: у значені напруги або струму, у амплітуді імпульсів, у часовому інтервалі між імпульсами, або у частоті їх слідування. Для того, щоб ЕОМ могла сприймати цю інформацію її потрібно перетворити в цифровий код. Тому обов'язковим елементом вимірювальної установки є один, або декілька аналого-цифрових перетворювачів (АЦП).

В автоматизованих вимірювальних установках ЕОМ виконує функцію керування процесом вимірювання. Програма, яка керує вимірювальною установкою містить інформацію про режими і послідовність вимірювань. ЕОМ повинна в потрібні моменти часу вмикати і вимикати вимірювальні пристрої, а при необхідності - змінювати їх параметри і умови експерименту. При виконані ЕОМ функцій керування виникає потреба у перетворені сигналів із цифрової форми в аналогову. Тому типовим елементом автоматизованих вимірювальних систем є також цифро-аналогові перетворювачі (ЦАП).

Швидкодія сучасних мікро-ЕОМ дозволяє виконувати попередній аналіз інформації, яка поступає від датчиків, в процесі її уведення в ЕОМ, або, як говорять, в реальному часі. Попередній аналіз даних робиться з метою відбору корисної інформації, а також для вироблення необхідних сигналів керування.

Існує два основних способи керування любими зовнішніми по відношенню до ЕОМ пристроями: режим програмного керування та режим переривань. Правильний вибір і суміщення цих режимів дуже важливо для ефективної роботи вимірювального комплексу. Крім того, при використані режиму переривань виникає необхідність у вирішені ряду методичних питань: призначення пристроям пріоритетів, розділення функцій обробки даних по рівнях переривань і інше. В цьому відношенні програмування апаратури в стандарті КАМАК має значну специфіку.

Розглянемо деякі типи ЦАП і АЦП, які використовуються в сучасних мікропроцесорних системах обробки даних.

Цифро-аналогові перетворювачі. Всі види ЦАП можна умовно розділити на дві групи: з прецизійними резистивними матрицями та безматричні ЦАП. В першій групі по способу формування сигналу розрізняють три типи схем: з додаванням струмів (мал. 6.1 *а*), з поділом напруг, з додаванням напруг. В мікроелектронному виконанні використовуються перших два типи. Із мікросхем другої групи можна назвати два типи ЦАП: з активними подільниками струмів та стохастичні.

Схема мал. 6.1 *а* відповідає структурній схемі мікросхеми 10-розрядного ЦАП КР572ПА1. Струм I0, величина якого задається напругою Uоп, послідовно ділиться у вузлах резистивної матриці (РМ) R-2R по двійковому закону:

$$I1=I0.2\text{-}1;\ I2=I0.2\text{-}2;\ In=I0.2\text{-}n.$$

Двійковий код числа, яке потрібно перетворити в аналогову форму, подається на входи X1, X2,...Xn. Коли на і-му вході високий рівень напруги (Xi=1) ключ Кі замикається так, що струм даного розряду РМ поступає на вихід 1. Коли Xi=0 цей же струм передається через ключ Кі на вихід 2. Якщо до виходів 1, 2 РМ приєднаний операційний підсилювач, то на аналоговому виході ЦАП буде напруга

$$Uвих=UопRoc(X1.2n\text{-}1+X2.2n\text{-}2+ \dots +Xn.20)/R.2n.$$

Вибираючи Uоп=10,24 В, для ЦАП КР572ПА1 Roc=R і n=10, одержимо:

$$Uвих=N.10\text{-}2\ В.$$

Де N - десятковий код двійкового числа, яке перетворюється в аналогову форму.

Час встановлення вихідної напруги (інтервал часу від подачі на вхід цифрового коду до моменту, коли вихідна напруга прийме задані межі) для ЦАП КР572ПА1 складає 5 мкс. Із швидкодіючих ЦАП, орієнтованих на використання в мікропроцесорних пристроях, слід назвати мікросхеми 12-розрядних ЦАП К594ПА1 та К1108ПА1 (час встановлення складає 0,4 мкс).

Аналого-цифрові перетворювачі. По структурі АЦП діляться на два типи: з використанням ЦАП і без них. До першого типу відносяться АЦП відслідковуючого, розгортаючого типів та порозрядного урівноваження.

На мал. 6.2 *а* приведена структурна схема АЦП послідовного порозрядного урівноваження. Двійковий код числа, яке відповідає цифрові формі представлення напруги Uвх, формується послідовно, розряд за розрядом в регістрі послідовного наближення 2. Коли на вхід схеми керування 1 подається сигнал З/П ("зупинка/перетворення") на виходах регістру 2 встановлюються нулі, а потім в старший розряд записується лог. 1. Одержане в регістрі 2 число, перетворюється ЦАП в напругу Uс, яка компаратором 3 порівнюється з вхідною напругою. Якщо виконується умова Uвх>Uс, то число, в яке перетворюється вхідна напруга, дійсно містить одиницю в старшому розряді, в іншому випадку в старший розряд регістру 2 записується лог. 0. Потім лог. 1 записується в наступний розряд регістру 2, знову Uс порівнюється з Uвх і вирішується, чи зберігати лог.1 в цьому розряді, чи ні. Таким чином перевіряються всі розряди. Потім, сформований в регістрі 2 код, через вихідні буферні схеми (ВБ) виводиться на шину даних. Одночасно тригер 4 формує сигнал ГОТ ("дані готові"). На мал. 6.2б приведені часові діаграми роботи АЦП.

Прикладом АЦП послідовного порозрядного урівноваження, який орієнтований на роботу в складі мікропроцесорних систем є мікросхема К1113ПВ1 (мал. 6.3). Ця мікросхема має 10-розрядну шину даних з Z-станом, що значно спрощує її узгодження з шиною даних мікропроцесора.

Аналого-цифровий перетворювач К1113ПВ1 може перетворювати однополярну напругу в межах 0...10 В (вивід 15 повинен бути приєднаний до спільного провідника) і двополярну напругу в межах -5...+5 В (вивід 15 залишається вільним, ніяка напруга на нього не подається). Коли на 11 виводі мікросхеми напруга високого рівня (лог. 1) цифрові виходи 9 - 1, 18 знаходяться в Z-станах, а регістр послідовного наближення (див. мал. 6.2) - у вихідному стані. При зміні напруги на виводі 11 з високого рівня на низький розпочинається перетворення вхідної напруги у цифровий код. Протягом часу перетворення (не більше 30 мкс) на виводі 17 зберігається лог. 1. Після закінчення процесу перетворення на цьому виводі встановлюється лог. 0, відкриваються вихідні буферні схеми і на цифрових виходах встановлюється код результату перетворення. Такий стан цифрових виходів АЦП зберігається доти, доки сигнал на виводі 11 (“зупинка/перетворення”) не зміниться з низького рівня на високий. Після цього шина

даних АЦП переводиться в Z-стан, а схема керування АЦП готується до наступного циклу перетворення. Новий цикл перетворення може розпочинатись не раніше, як через 2 мкс. Мікросхема має два виводи для приєднання спільних провідників (“землі”) - аналогового (вивід 14) і цифрового (вивід 16). Напруга між цими виводами не повинна перевищувати 0,2 В. Резистори R1 - R4 дозволяють регулювати напругу зміщення нуля АЦП. При роботі АЦП в режимі перетворення двополярної напруги резистор R2 не ставиться. Резистор R5 дозволяє регулювати напругу в кінцеві точці шкали в межах $\pm$ 3 МР.

В інтегральному виконані випускається спеціалізована ВІС КР572ПВ1 на базі якої можна реалізувати аналого-цифрове і цифро-аналогове перетворення сигналів в складі мікропроцесорної системи з організацією по-байтового обміну даними.

Завдання для домашньої підготовки.

1. Ознайомтесь з структурою мікропроцесорної вимірювальної системи.
2. Ознайомтесь з основними методами цифро-аналогового та аналого-цифрового перетворення сигналів.
3. Вивчіть програму 6.1, яка забезпечує обмін даними між ЕОМ і аналого-цифровим перетворювачем (вольтметр Ф-294) та цифро-аналоговим перетво-рювачем.

Завдання до лабораторної роботи.

Завдання 1. Вивчення структури та програмного забезпечення мікропроцесорної вимірювальної системи на лабораторному макеті. Порядок виконання завдання:

- увімкніть живлення двокоординатного потенціометра та цифро-аналогового перетворювача;
- завантажте з магнітофону (або ROM-диску) програми: драйвери АЦП і ЦАП (програма 6.1) в машинних кодах та на асемблері, управляючу програму EXPEREMENT, написану на мові BASIC;
- здійсніть пуск програми EXPEREMENT директивою RUN;
- працюйте під керуванням програми;

Програма 6.1

```
            ; Драйвери ЦАП і АЦП
            ;
BUFU        EQU 0B4DCH
NOMZ        EQU 0B4DDH
FUNK        EQU 0B4DEH
BNAP        EQU 0B4E0H
BSTR        EQU 0B4E2H
KEY         EQU 0F803H
            ORG 0B400H
START:      MVI A, 80H      ; ініціалізація ЦАП
            OUT 0F3H
            MVI A, 0
            OUT 0F2H
            OUT 0F0H
            MVI A, 57H
            OUT 0F1H
            MVI A, 0
            OUT 0F1H
            RET
WZAP:       LXI H, BUFU     ; запис коду в регістр ЦАП
            MOV A, M
            OUT 0F0H
            INX H
            MOV A, M
            OUT 0F1H
            MVI A, 0
            OUT 0F1H
            RET
ZAPF:       LXI H, FUNK     ; запис коду функції
            MOV A, M
            OUT 0F1H
            ANI 0A8H
            OUT 0F1H
            MVI A, 0
            OUT 0F1H
            RET
NSTR:       MVI A, 92H      ; ініціалізація портів
            OUT 0C3H        ; уведення
            MVI A, 0BBH
            OUT 0C2H
            RET
NAPR:       CALL NSTR       ; читання коду напруги
            MVI A, 40H      ; з АЦП
            STA FUNK
            CALL ZAPF
            CALL DEL
            LXI H, BNAP
            CALL READ
            RET
STRR:       CALL NSTR       ; читання коду струму
            MVI A, 0C0H     ; з АЦП
```

```
        STA FUNK
        CALL ZAPF
        CALL DEL
        LXI H, BSTR
        CALL READ
        RET
DEL:    PUSH H          ; затримка на 100 мс
        PUSH PSW
        LXI D, 2400H
M1:     DCX D
        MOV A, D
        ORA E
        JNZ M1
        POP PSW
        POP H
        RET
READ:   MVI B, 8BH      ; уведення даних
        MVI C, 40H      ; з портів
        LDA FUNK
        RRC
        RRC
        ORA B
        OUT 0C2H
        CALL DEL
        ORA C
        OUT 0C2H
        CALL DEL
        IN 0C0H
        MOV M, A
        INX H
        IN 0C1H
        ANI 1FH
        MOV M, A
        MVI A, 9BH
        OUT 0C2H
        RET
PROP:   CALL KEY        ; контроль клавіші
        CPI 20H         ; <пропуск>
        JNZ PROP
        RET
        END
```

Завдання 2. На основі програм 6.1 розробіть програму, яка уводить дані з АЦП (вольтметр Ф-294) в ЕОМ і виводить результат вимірювання на дисплей ЕОМ (програма 6.2).

При виконанні завдання використайте програму HELP, яка містить підпрограми керування і обробки даних.

Керування запуском АЦП на перетворення сигналу та уведенням коду напруги (або струму) у пам’ять ЕОМ здійснюється підпрограмами програми 6.1: NSTR, NAPR, STRR, READ (див. таблицю 6.1). Інформація з АЦП уводиться через порти А і В ППІ КР580ВВ55, виводи якого знаходяться на з’єднувачі Х2 (ВНЕШ.2) ЕОМ ПК-01 “Львів”

Таблиця 6.1. Адреси буферів та підпрограм програми 6.1.

Адреса H-код D-кодМнемонікаПризначенняПримітка
B4DC B4DD B4DE B4E0 B4E1 B4E2 B4E3 B400 B413 B422 B431 B43A B44F B464 B472 B495 46300 46301 46302 46304 46305 46306 46307 46080 BUFU NOMZ FUNK BNAP BSTR START WZAP ZAPF NASTR NAPR STRR DEL READ PROP Буфер коду напруги Буфер номеру ЦАП Буфер коду функції Буфер даних напруги Буфер даних напруги Буфер даних струму Буфер даних струму Підпрограма ініціалізації ЦАП Підпрограма запису коду в ЦАП Підпрограма запису коду функції Підпрограма ініціалізації портів уведення та АЦП Підпрограма читання коду напруги з АЦП Підпрограма читання коду струму з АЦП Підпрограма затримки на 100 мс Підпрограма уведення даних з портів Підпрограма контролю клавіші <пропуск> Мол.байт Ст. байт Мол.байт Ст. байт

Результат вимірювання напруги на виході вольтметра представляється двійково-десятковим кодом. Через порт А уводиться два молодших розряди десяткового коду, а через порт В - два старших розряди. Порт С використовується для керування вольтметром. Призначення ліній інтерфейсу АЦП дано в таблиці 6.2.

Таблиця 6.2. Адреси портів та призначення ліній інтерфейсу АЦП.

ПортАдресаЛінії портуСигнал АЦП
PA0C0HA0 A1 A2 A3 A4 A5 A6 A71.10-2 2.10-2 4.10-2 8.10-2 1.10-1 2.10-1 4.10-1 8.10-1PB0C1HB0 B1 B2 B3 B41.100 2.100 4.100 8.100 1.101PC0C2HC5 C6 C7Керування комутатором Запуск АЦП Керування індикаторами

Завдання 3. Написати програму керування цифро-аналоговим перетворювачем. На основі програм 6.1 та HELP написати програму керування пером двокоординатного потенціометра (програма 6.3):

- підняти перо;
- перемістити перо в точку з координатами X,Y;
- опустити перо.

Цифро-аналоговий перетворювач виготовлений на основі мікросхем КР572ПА1, представляє собою програмовний пристрій виведення інформації з ЕОМ і служить для перетворення інформації, представленої цілим числом від 0 до 255, в постійну напругу від 0 до 10 В. Пристрій містить два канали цифро-аналогового перетворення (ЦАП-1 і

ЦАП-2). Інформація в ЦАП виводиться із ЕОМ через порт А інтерфейсу ЦАП, який реалізований теж на основі мікросхеми КР580ВВ55. Програмування різних функцій цифро-аналогового перетворювача (див. таблицю 6.3) здійснюється через порт В. Порт С використовується для програмування режимів вихідних буферів інтерфейсу.

Таблиця 6.3. Коди функцій цифро-аналогового перетворювача

Номер Код функції H-код D-кодПризначення1 2 3 4 5 6 7 81 2 04 4 0C 12 10 16 30 48 C0 192 40 64Строб даних в регістр ЦАП-1 Строб даних в регістр ЦАП-2 Встановити "+" вихідної напруги ЦАП-1 Встановити "-" вихідної напруги ЦАП-1 Встановити "+" вихідної напруги ЦАП-2 Встановити "-" вихідної напруги ЦАП-2 Замкнути реле (опустити перо) Розімкнути реле (підняти перо)

З точки зору програміста пристрій ЦАП представлений п’ятьма регістрами: регістр даних ЦАП-1 - 8 біт; регістр даних ЦАП-2 - 8 біт; регістр знаку ЦАП-1 - 1 біт; регістр знаку ЦАП-2 - 1 біт; регістр реле - 1 біт.

Адреса порту А - 0F0H; порту В - 0F1H; порту С - 0F2H. Адреса регістру слова керування інтерфейсу ЦАП - 0F3H

Драйвер ЦАП складається з трьох підпрограм START, WZAP, ZAPF (див. таблицю 6.1).

Програма 6.4

```
190 REM ------- Програма HELP ----------------------
200 REM ------ запис коду напруги в ЦАП-1 ------
202 REM  вхідний параметр: X - код напруги
210 POKE 46300,X
220 POKE 46301,1
230 POKE 73,19
240 GOSUB 300
250 RETURN
260 REM ------ підпрограма вимірювання напруги -----
270 POKE 73,58
280 GOSUB 300
290 RETURN
300 REM ------ перехід в асемблер ------
310 POKE 74,180
320 X=USR(X)
330 RETURN
340 REM ------ підпрограма вимірювання струму ------
350 POKE 73,79
360 GOSUB 300
370 RETURN
400 REM ------ підпрограма перетворення коду ------
402 REM  вихідний параметр:
```

```
403 REM  значення напруги V і струму I
405 R=30
410 B=2
420 GOSUB 500
430 H=P*10+O+D*0.1+S*0.01
440 B=0
460 GOSUB 500
470 Q=P*10+O+D*0.1+S*0.01
480 V=Q-H
490 I=(H/R)*1000
500 Z=PEEK(46304+B)
510 D=INT(Z/16)
520 S=Z-D*16
530 Y=PEEK(46305+B)
540 P=INT(Y/16)
550 O=Y-P*16
560 RETURN
```

Зміст звіту

Звіт повинен містити:

- назву та мету роботи;

- структурну схему мікропроцесорної вимірювальної системи та її опис;

- часові діаграми роботи вольтметра Ф-294 в режимі аналого-цифрового перетворення;

- лістинг програм 6.2 та 6.3.

Додаткові завдання

Д6.1. Розробіть схему приєднання АЦП типу К1113ПВ1 до мікро-ЕОМ ПК-01 "Львів".

Д6.2. Розробіть схему приєднання ЦАП типу КР572ПА1 до мікро-ЕОМ ПК-01 "Львів".

Д6.3. Напишіть програму, яка забезпечує уведення інформації з АЦП К1113ПВ1 в мікро-ЕОМ.

Д6.4. Напишіть програму виведення інформації з ЕОМ на екран осцилографа.

Д6.5. Напишіть програму, яка формує у пам’яті ЕОМ масив даних з N вимірювань напруги і струму.

Контрольні запитання

1. Поясніть структуру мікропроцесорної вимірювальної системи.
2. Яку роль відіграє мікро-ЕОМ в структурі вимірювальної системи?
3. Що називають аналого-цифровим перетворенням даних?
4. Які методи аналого-цифрового перетворення даних Ви знаєте?
5. Що називають цифро-аналоговим перетворенням даних?
6. Які методи цифро-аналогового перетворення даних Ви знаєте?
7. Яку програму називають драйвером ЦАП? Поясніть алгоритм її роботи.
8. Яку програму називають драйвером АЦП? Поясніть алгоритм її роботи.
9. В якій частині програми 6.1 мікро-ЕОМ записує дані в регістр даних ЦАП?
10. В якій частині програми 6.1 мікро-ЕОМ читає дані з вольтметра?

Лабораторна робота № 7

ОСНОВИ ПРОГРАМУВАННЯ СИСТЕМИ КАМАК

Мета роботи. Вивчення основних принципів та стандартів системи модульної електроніки КАМАК; ознайомлення із способами програмування функціональних модулів та апаратури КАМАК.

Короткі відомості з теорії

1.1. Система КАМАК

Міжнародна система модульної електроніки КАМАК забезпечує обмін інформацією між вимірювальними приладами і пристроями, які використовуються в експерименті, та цифровими системами обробки даних. Система КАМАК, є інтерфейсом зв'язку між вимірювальною апаратурою і ЕОМ. На початку вона була орієнтована для потреб спеціалістів в області ядерної фізики, але швидко завоювала популярність в різних областях експериментальних досліджень і практично з моменту її створення отримала статус міжнародного стандарту на апаратуру для наукових досліджень.

Основними особливостями системи КАМАК є :

- модульний принцип побудови, що забезпечує можливість створення на її основі інформаційно-вимірювальних систем;
- конструктивна однорідність системи, яка досягнута шляхом уніфікації конструкцій, включаючи крейт для розміщення функціональних модулів;
- магістральна структура інформаційних зв’язків між функціональними модулями;
- широке використання принципів програмного керування.

В залежності від складності задач, які вирішуються вимірювальними системами вони можуть мати різну кількість функціональних модулів. Модулі розміщуються у крейті - спеціальному каркасі, який має 25 місць для встановлення модулів. Два крайніх правих місця у крейті завжди займає контролер крейту - спеціальний модуль, що керує роботою інших модулів і забезпечує обмін інформацією між модулями і ЕОМ. В самому крейті обмін інформацією між контролером і модулями відбувається по сукупності провідників - магістралі крейту.

В організації системи КАМАК (мал. 7.1) є певна аналогія з структурною організацією мікропроцесорних систем. Для обох систем характерним є магістральний принцип передачі інформації та принцип відкритої архітектури, а також структура самої магістралі - наявність шини даних, шини адреси і керуючої шини.

Основні принципи КАМАК закладені в трьох стандартах - механічному, електричному та логічному.

1.2. Стандарти системи КАМАК

Механічний стандарт визначає розміри основних конструктивних елементів системи - крейта та модулів. Одним з основних конструктивних елементів системи КАМАК являється крейт. Конструкція крейта стандартом повністю не визначена. Задані лише основні внутрішні розміри його вікна. Крейт містить 25 верхніх та нижніх напрямних, по яких в нього вставляються функціональні модулі. Напрямні розміщені з кроком 17,2 мм. Обов'язковим елементом конструкції крейта є 86-контактні з'єднувачі. Пара напрямних і з'єднувач утворюють *станцію* крейта - тобто місце для встановлення функціонального модуля. Станціям крейту присвоєнні номери від 1 до 25, які рахуються зліва на право. Внизу біля кожної станції знаходиться отвір з різьбою для фіксуючого гвинта функціонального модуля.

Основою механічної конструкції функціонального модуля є пара напрямних, які приєднані до передньої і задньої панелей. Передня панель служить для кріплення пристроїв комутації (з'єднувачі, вимикачі), керування та відображення. Внизу на передній панелі кріпиться фіксуючий гвинт. Мінімальна ширина передньої панелі модуля 17,2 мм. Модулі можуть мати і більшу ширину, але кратну 17,2 мм. Для змінних блоків, які мають ширину більшу за розмір однієї станції, дозволяється використовувати декілька напрямних. В модулі знаходиться монтажна плата на якій монтується електрична схема. На задній частині монтажної плати знаходяться контакти (2х43) для з'єднання модуля із магістраллю крейта.

Електричний стандарт визначає всі електричні величини в системі КАМАК. Стандартом передбачені певні значення напруг джерел живлення, максимальні навантаження по струму джерел живлення та контактів з'єднувачів, максимальні потужності окремих модулів, рівні вхідних і вихідних сигналів функціональних модулів, а також рівні і часові характеристики сигналів магістралі крейта.

В крейті є обов'язковими такі значення напруг джерел живлення: ±6 В, з максимальним струмом 25 А - для живлення мікросхем; ±24 В, з максимальним

струмом 6 А - для живлення транзисторів і мікросхем, з підвищеною напругою живлення.

До додаткових джерел живлення відносяться: ±12 В, з максимальним струмом 2 А - для живлення транзисторів; +200 В, з максимальним струмом 0,1 А - для живлення газорозрядних індикаторів; ~117В, з максимальним струмом 0.5 А - для живлення двигунів та інших пристроїв.

В системі КАМАК в більшості випадків для передачі цифрових сигналів використовується від'ємна логіка, тобто логічній 1 відповідає низький, а логічному 0 високий потенціал.

В системі КАМАК прийнято, що передача наносекундних сигналів має передаватися по лініях зв'язку з хвильовим опором рівним 50 Ом. Зміну рівнів логічних сигналів на виходах і входах модулів прийнято виражати в цьому випадку в одиницях струму, а не в одиницях напруги.

Логічний стандарт. Цей стандарт визначає порядок обміну інформацією лише між функціональними модулями і контролером крейта. Він не поширюється на порядок взаємодії контролера крейта з ЕОМ. Стандартом повністю визначені функціональне призначення сигналів на магістралі крейта. Взаємодія функціонального модуля з контролером через магістраль крейта зводиться до трьох видів операцій:

- пересилка інформації по шині даних магістралі крейту від функціонального модуля до контролера, або від контролера до модуля;

- керування роботою окремих функціональних схем всередині модуля сигналами, що поступають від контролера;

- перевірка контролером стану окремих функціональних схем модуля.

Сигнали магістралі крейту можна розділити на 4 групи: сигнали команди КАМАК (N, A, F), сигнали стану (X, Q, L, B), сигнали керування (Z, C, I, S1, S2) та сигнали передачі даних (R, W)

Сигнали команди КАМАК (N, A, F). Сигнал N служить для передачі номеру або адреси станції в крейті. В крейті кожний функціональний модуль адресується по окремих лініях від контролера. Для звертання до функціональних модулів, які встановлюються на певних станціях крету, використовуються адреси від 1 до 23 (24 та 25 місця займає контролер крейту). Згідно стандарту після літери, якою позначається сигнал, в круглих дужках десятковим числом записується номер станції. Наприклад, запис N(7) означає звертання до функціонального модуля, який встановлений на сьомій станції крейту.

Сигнал A - субадреса, тобто підмножина адрес, які присвоюються самостійним функціональним частинам всередині модуля. Наприклад, якщо модуль має декілька цифро-аналогових перетворювачів, то запис інформації в регістри окремих ЦАП буде здійснюватись по різних субадресах. Стандартом передбачена можливість використання 16 субадрес. Вони позначаються A(0), ... A(15). Декодування адреси здійснюється всередині модуля.

Сигнал F - операція, або функція КАМАК. Код операції, який поступає від контролера до функціонального модуля, повністю визначає набір дій, які повинні бути виконані в модулі. Стандартом передбачено використання 32 операцій. Вони позначаються: F(0), ... F(31). Коди функції розділяються на три групи: стандартні коди, яким відповідають визначені стандартом дії в модулях і контролерах (18 кодів); резервні коди - для подальшого їх використання при розширенні системи КАМАК (6 кодів); нестандартні коди (8 кодів), використання яких стандартом не регламентується.

Сигнали стану (X, Q, L, B). Сигнал X - *Команда прийнята*. Цей сигнал є обов'язковою відповіддю модуля на будь-яку адресовану йому команду NAF. Кожний модуль повинен послати сигнал X контролеру, якщо прийняв команду і може її виконати.

Сигнал Q - *Відповідь*. На любу адресну операцію (команду, яка містить адресні сигнали N і A) модуль може (проте не обов'язково) відповісти сигналом Q=1 або Q=0. Значення сигналу Q повинні бути чітко визначені для конкретної операції і конкретної частини електричної схеми модуля при його розробці. За допомогою сигналу Q можна отримати інформацію про стан деякої частини функціонального модуля.

Сигнал L - *Запит на обслуговування*. Любий функціональний модуль, який встановлений у крейт, може послати в контролер сигнал L. Цим сигналом він повідомляє контролеру про необхідність виконати певні дії по обслуговуванню модуля. Модуль виробляє сигнал L коли він готовий до обміну інформацією з контролером. Незалежно від числа функціональних вузлів в модулі, які одночасно виставили запит на обслуговування, він посилає контролеру завжди лише один сигнал L. Порядок і першочерговість обслуговування запитів модуля визначає експериментатор, котрий програмує роботу своєї вимірювальної системи.

Сигнал B - *Зайнято*. Цей сигнал виробляє контролер крейту. Сигнал B супроводжує любі сигнали, які передає контролер по магістралі. Значення B=1 повідомляє всім функціональним модулям про те, що магістраль в даний момент зайнята контролером для виконання певної команди.

Сигнали керування (Z, C, I, S1, S2). Сигнал Z - *Початкове встановлення*. Він служить для встановлення у вихідний стан електричних схем функціональних модулів. Сигнал Z являється першим сигналом, який виробляє контролер згідно команди від ЕОМ на початку виконання програми. Цей сигнал, як правило, виробляється і контролером, коли вмикається живлення крейту. По сигналу Z регістри функціональних модулів приймають визначений початковий стан, всі запити на обслуговування в модулях анулюються і забороняється передача сигналів L з модулів на магістраль крейту.

Сигнал C (Clear) - *Очищати*. Сигнал C=1 використовується для того, щоб при потребі встановити тригери і регістри функціональних модулів в крейті у вихідний стан. Цей сигнал, як і сигнал Z, виробляє контролер згідно команди від ЕОМ. Різниця в дії сигналів полягає в тому, що сигнал Z генерується контролером лише один раз на початку роботи, а сигнал C може використовуватися багато разів в процесі керування роботою модулів при виконанні програми. Крім того, сигнал Z обов'язково повинен використовуватись при розробці електричної схеми модуля, а сигнал C може не використовуватися в модулі, оскільки встановлення тригерів модуля у вихідний стан може бути виконане спеціальними командами.

Сигнал I - *Заборона*. Сигнал I може забороняти любі дії в модулі, заборона яких з допомогою цього сигналу була передбачена при розробці модуля. Цей сигнал може вироблятися як контролером крейта програмно по спеціальній команді від ЕОМ, так і функціональним модулем, якщо генерація цього сигналу в модулі була передбачена апаратно.

Сигнали S1, S2 - *Строб S1* і *Строб S2*. Ці сигнали служать для стробування, тобто для виконання визначених дій в функціональних модулях, або у контролері крейта у певні моменти часу. Під час дії сигналу S1 в модулях або в контролері виконуються любі дії, які не змінюють стан сигналів на інформаційних шинах. Любі дії в модулях або в контролері, які можуть привести до зміни сигналів на інформаційних шинах магістралі повинні викликатись сигналом S2.

Сигнали передачі даних (R, W). В крейті КАМАК передбачена можливість обміну даними по лініях R та W. Лінії R - для читання. Дані з функціонального модуля передаються контролеру, тобто контролер читає (Ready) дані. Лінії W - для запису. Через ці лінії з контролера в функціональний модуль передаються дані. Інформація, яка передається по лініях R і W, може бути даними, які отримані в процесі вимірювань, а

також містити і додаткову інформацію, наприклад, про стан окремих вузлів модуля або зовнішніх приладів, приєднаних до крейту.

1.3. Магістраль крейту

Обмін інформацією та передача сигналів керування між контролером і функціональними модулями здійснюється по магістралі крейту. *Магістраль крейту представляє* канал зв'язку, окремі лінії (провідники) якого певним способом сполучають контакти з'єднувачів окремих станцій (мал. 7.2). Контакти перших 24 з'єднувачів магістралі сполучені однаково. Це з'єднувачі так званих нормальних станцій. Станцію крейту, яка відповідає 25-му з'єднувачу, називають керуючою. Сполучення ліній магістралі з контактами з'єднувача цієї станції відмінні від решти станцій.

По типу сполучень лінії магістралі крейту бувають: *індивідуальні*, це такі котрі сполучають певний контакт нормальної станції з певним контактом керуючої станції; *наскрізні* - сполучають одні і ті ж контакти з'єднувачів всіх 25-и станцій (виключення складають лінії R і W, які не сполучені з контактами 25-ї станції)

Шини магістралі крейту. По функціональному призначенню лінії магістралі поділяють на сім шин.

Шина адреси. Адресна шина містить: 24 індивідуальних лінії по яких одиничним позиційним кодом передається номер станції N та 4 наскрізних лінії для передачі двійкового коду субадреси A. Ці лінії позначаються A1, A2, A4, A8. Цифри після літери вказують вагу розряду двійкового коду.

Шина операцій. Ця шина містить 5 наскрізних ліній, по яких від контролера до функціональних модулів передається двійковий код операції (команди) F. Лінії позначаються F1, F2, F4, F8, F16. Цифри після літери теж вказують вагу розряду двійкового коду.

Шина стану. По лініях цієї шини передаються сигнали, які несуть інформацію про стан функціональних модулів або крейту. Ця шина містить: 24 індивідуальних лінії L, по яких від модуля до контролера передаються сигнали запиту на обслуговування та 3 наскрізні лінії для передачі сигналів X, Q, B.

Шина керування. Ця шина містить 5 наскрізних ліній для передачі сигналів керування Z, C, I, S1, S2.

Шина даних. Шина даних складається із 24 наскрізних ліній для передачі даних у модулі для запису (лінії запису W) та із 24 наскрізних ліній для передачі даних із модулів

до контролера (лінії читання R). Лінії для запису позначають W1, ... W24, а лінії для читання - R1, ... R24.

Шина живлення. Для забезпечення нормальної роботи електричних схем функціональних модулів та контролера на магістралі крейту передбачено 14 наскрізних ліній живлення. Із них 6 ліній стандартизували для обов'язкових номіналів напруг, 6 ліній призначені для додаткових джерел живлення. Для подальшого розвитку зарезервовані ще 2 лінії живлення.

Шина вільних ліній. З метою розширення функціональних можливостей модулів крейту магістраль містить наскрізні вільні лінії (2 лінії), які можуть використовуватись довільно, а з'єднувачі станцій мають вільні контакти.

Характеристики сигналів магістралі крейту. В системі КАМАК на магістралі використовується синхронний цикл обміну інформацією. Це означає, що часові характеристики сигналів (тривалість і затримки в часі одних сигналів по відношенню до інших) чітко визначені і завжди однакові. Сигнали, які визначають характер дій мають тривалість 1 мкс. Сигнали стробуючі S1 і S2 мають тривалість 0,2 мкс. Ці сигнали повинні появлятися на лініях з певною затримкою по відношенню до початку часового циклу. Це обумовлено тим, що до появи сигналів S1 і S2 сигнали на інших лініях магістралі повинні досягти рівнів логічної 1. Стандартом передбачено, що тривалість перехідних процесів на всіх лініях магістралі не повинна перевищувати 0,1 мкс, а час затримки поширення сигналів електричними схемами модулів не повинен перевищувати 0,3 мкс.

На мал. 7.3 приведені сигнали часового циклу КАМАК на лініях магістралі крейту при виконані командної операції. Часові інтервали, які обумовлені перехідними процесами на малюнку не показані. Пунктирні лінії вказують допустимий діапазон часу затримки сигналів відносно початку циклу. Для сигналів B, N, A, F ця затримка не повинна перевищувати 50 нс, а для сигналів даних (R, W) і стану
(Q, X, I) - 300 нс.

При виконанні часового циклу КАМАК першим в часі появляється сигнал зайнятості B. Потім виробляються сигнали N, A, F. Якщо модуль прийняв команду і дешифрував її, то він виробляє сигнал К*оманда прийнята* X. Контролер аналізує стан сигналу X в момент, коли появляється сигнал стробування S1. Значення сигналу X повинно зберігатися незмінним до появи сигналу S2.

Процеси, обумовлені операцією F, в модулі повинні розпочинатися в момент появи стробуючих сигналів S1 або S2. Стандартом передбачено, що сигнал S1 використовується тоді, коли виконуються операції, які не приводять до зміни логічних сигналів на лініях даних. По сигналу S2 виконуються операції, які приводять до зміни стану ліній R, W. Запис інформації в модуль з ліній запису W і запис даних в контролер з ліній читання R повинні здійснюватись в момент появи стробуючого сигналу S1. Сигнал Q, який виробляється модулем, повинен сприйматися контролером по S1.

Сигнали запиту на лініях L магістралі можуть появлятися в довільні моменти часу. Проте, модуль, який виставив сигнал запиту на обслуговування (L=1), повинен зняти його, якщо він приймає від контролера команду.

1.4. Операції КАМАК

По призначенню всі операції КАМАК поділяють на чотири групи: операції читання; перша група операцій керування; операції запису; друга група операцій керування. Повний список операцій приведений в таблиці 7.1. При пояснені дій, які відбуваються в модулях в результаті виконання операцій, використовують терміни - регістри першої та другої групи. До регістрів першої групи відносять робочі регістри, які поряд з іншими елементами електричної схеми модуля забезпечують виконання модулем основних функцій. Регістри другої групи - це допоміжні регістри, які зберігають різну службову інформацію.

Операції читання F(0) - F(3) здійснюють читання інформації з модуля по лініях R. Інформація з регістрів модуля появляється на лініях R при виконанні часового циклу КАМАК і записується в регістри контролера. Вміст регістрів модуля не змінюється (при операціях F(0), F(1), F(3)). У випадку, коли виконується операція F(2) за один цикл реалізується дві дії - читання даних з регістру і його очищення (запис логічного 0 в тригери регістра).

Таблиця 7.1. Операції КАМАК

Номер п/пПризначення операціїФункція FДвійковий код операції0 1 2 3Читання з регістру 1-ї групи Читання з регістру 2-ї групи Читання та очищення регістру 1-ї групи Читання інверсного коду з регістру 1-ї групи0 1 2 300000 00001 00010 000114 5 6 7Нестандартна Резервна Нестандартна Резервна4 5 6 700100 00101 00110 001118 9 10 11Перевірка запитів Очищення регістра 1-ї групи Очищення запиту Очищення регістра 2-ї групи8 9 10 1101000 01001 01010 0101112 13 14 15Нестандартна Резервна Нестандартна Резервна12 13 14 1501100 01101 01110 0111116 17 18 19Запис в регістр 1-ї групи Запис в регістр 2-ї групи Вибірковий запис в регістр 1-ї групи Вибірковий запис в регістр 2-ї групи16 17 18 1910000 10001 10010 1001120 21 22 23Нестандартна Вибіркове очищення регістра 1-ї групи Нестандартна Вибіркове очищення регістра 2-ї групи20 21 22 2310100 10101 10110 1011124 25 26 27Заборона Виконання Дозвіл Перевірка статусу24 25 26 2711000 11001 11010 1101128 29 30 31Нестандартна Резервна Нестандартна Резервна28 29 30 3111100 11101 11110 11111

Операції запису F(16) - F(19), F(21), F(23) здійснюють запис інформації з ліній W в регістри модуля, субадреса яких вказана в команді КАМАК. Операції вибіркового запису F(18), F(19) записують логічну 1 лише у певні розряди регістрів першої або другої груп. Вибір розрядів, у які буде записана логічна 1 здійснюється за допомогою розрядів слова даних, яке передається в цьому ж циклі по лініях W. Слово даних мусить містити логічні 1 в тих розрядах, у яких в регістрах модуля потрібно встановити 1. Логічні 0 в розрядах слова даних не змінюють значення розрядів регістрів. Операції вибіркового очищення F(21), F(23) встановлюють логічний 0 в тих розрядах регістрів модуля, в яких розрядах слова даних, яке передається в цьому ж циклі по лініях W, записані логічні 1.

Операції F(8) - F(11) відносяться до першої групи операцій керування. Операція F(8) служить для контролю запитів на обслуговування по лініях L в модулі. Конкретна частина електричної схеми модуля, яка контролюється, вибирається згідно присвоєної їй субадресі. На операцію F(8) модуль обов'язково повинен дати відповідь контролеру по лінії Q. Операція F(10) використовується для встановлення тригерів в нуль, які формують, або запам'ятовують сигнали запиту на обслуговування. Операції F(9) та F(11) служать для запису логічного 0 в тригери регістрів першої та другої груп (очищення регістрів).

Друга група операцій керування F(24) - F(27) використовується для реалізації різних дій над окремими частинами електричної схеми модуля, котрі не потребують передачі інформації по лініях R і W. Операція F(24) використовується для заборони певних дій в модулі, або передачі певних сигналів. Операція F(26) дозволяє ті дії в модулі, які заборонені по F(24). Операція F(25) дозволяє або забороняє любі дії в модулі, лише в тих випадках, коли недоцільно використовувати операції F(24) або F(26). Операцією F(27) перевіряють стан будь яких функціональних частин електричної схеми модуля (при умові, що ця перевірка підтримана апаратно). На операцію F(27) модуль обов'язково повинен дати відповідь контролеру по лінії Q.

2. Програмування КАМАК.

В залежності від задач, які вирішуються, а також від програмних і апаратних засобів, які використовуються, можна виділити два напрямки в розробці програмного забезпечення системи КАМАК:

- використання спеціальних мов програмування КАМАК високого рівня;

- безпосереднє програмування апаратури КАМАК на мові Асемблера.

Використання спеціальних мов програмування КАМАК високого рівня дозволяє значно спростити процес програмування. Програмісту немає потреби детально

знайомитись з структурою апаратних засобів узгодження системи КАМАК з ЕОМ. При написані програм він використовує лише оператори та інші конструкції мови програмування, які описують виконання команд КАМАК.

Програмування апаратури КАМАК на мові Асемблера (програмування на “фізичному рівні”) вимагає детального вивчення логічної організації зв’язку контролера крейту з ЕОМ, ситеми переривань даної ЕОМ та її взаємодії із зовнішніми пристроями. Тому написання програм на асемблерному рівні є набагато складнішим і трудомістким, порівняно з програмуванням на мовах високого рівня. Як правило, програмування на фізичному рівні використовується тоді, коли вирішуються задачі організації керування вимірювальними процесами в реальному маштабі часу.

Апаратура КАМАК по відношеню до ЕОМ є нестандартним зовнішнім пристроєм, конфігурація якого змінюється у відповідності до вимог експерименту. Кожний раз, коли змінюється склад функціональних модулей вимірювальної системи, програмне забезпечення повино бути частково модифіковане або повністю замінене. Тому, розробляючи програми керування функціональними модулями КАМАК, які входять до складу вимірювальної системи, експериментатор повинен знати структурну організацію і загальний принцип роботи контролера крейту.

Логічна структура контролера крейту для різних ЕОМ стандартом КАМАК не визначена. В даній роботі описується структура і принцип роботи контролера типу ФК4410 виробництва Вільнюского заводу електровимірювальної техніки (VEMTG), який служить для зв’язку з ЕОМ типу PC XT/AT фірми IBM.

Контролер ФК4410 взаємодіє з ЕОМ по паралельному інтерфейсу через інтерфейсну плату, яка вставляється в з’єднувач (slot) системної шини ЕОМ (мал. 7.4). З точки зору програміста контролер представляє собою 13 вісьмирозрядних регістрів (табл.7.2), в частину з яких можна записувати інформацію, а з частини регістрів інформація лише читається.

Регістри контролера ФК4410, в які інформація лише записується:

регістр 0 (WH) - старший байт слова КАМАК, яке записується;

регістр 1 (WM) - середній байт слова КАМАК, яке записується;

регістр 2 (WL) - молодший байт слова КАМАК, яке записується;

регістр 3 (A) - субадреса модуля КАМАК, який адресується;

регістр 4 (F) - код функції, яка виконується;

регістр 5 (N) - номер станції крейту (1 ... 23), яка адресується;

регістр 6 - регістр керування; якщо біти Z або C встановлені в одиницю, то автоматично очищуються регістри N, A, F (регістри 5, 3, 4 табл. 7.2). Після запису в регістр керування бітів Z, C, I потрібно ініціювати цикл КАМАК, щоб контролер виробив відповідні сигнали на магістралі крейту. Біти AX1 - AX3 програмуються, якщо крейтом керують декілька контролерів.

Таблиця 7.2.

Номер регі-струПаралельний інтерфейс Адреса (відносна) Дані BA8 BA4 BA2 BA1 BD7 BD6 BD5 BD4 BD3 BD2 BD1 BD0 Коментар00000W24W23W22W21W20W19W18W17WH Дані, які записуються10001W16W15W14W13W12W11W10W9WM Дані, які записуються20010W8W7W6W5W4W3W2W1WL Дані, які записуються30011A8A4A2A1A Субадреса модуля40100F16F8F4F2F1F Функція50101N16N8N4N2N1N Номер станції60110AX4AX3AX2AX1ICZZ, C, I Біти70111Старт циклу КАМАК81000LL16L8L4L2L1XQLAM, X, Q91001R24R23R22R21R20R19R18R17RH Дані, які читаютьсяA1010R16R15R14R13R12R11R10R9RM Дані, які читаютьсяB1011R8R7R6R5R4R3R2R1RL Дані, які читаютьсяC1100AOALACСтатус

Регістр 7 - запис в цей регістр любих даних приводить до ініціювання циклу КАМАК. Наприклад, для виконання команди запису потрібно завантажити регістри N, A, F і дані W, а потім ініціювати цикл КАМАК шляхом запису в регістр 7 будь-яких даних.

Регістри контролера ФК4410, з яких інформація лише читається:

регістр 8 - читається інформація про результат виконання останнього циклу КАМАК (біти X і Q) та двійкових код (біти L16 - L1) станції з найвищим пріоритетом, яка виробила сигнал запиту на обслуговування. Біт L=0, якщо в крейті встановлений хоча б один сигнал *Запит.*

регістр 9 (RH) - старший байт слова КАМАК, яке читається;

регістр A (RM) - середній байт слова КАМАК, яке читається;

регістр B (RL) - молодший байт слова КАМАК, яке читається;

регістр C - статусний регістр контролера: біт AC приймає значення лог. 0, при ініціюванні циклу КАМАК і приймає значення лог. 1 після закінчення циклу КАМАК; біт A0 приймає значення лог. 1, якщо контролер працює в режимі ON LINE.

Приклад послідовності програмування регістрів контролера ФК4410 при організації обміну даними з ЕОМ.

ОПЕРАЦІЯ ЗАПИСУ ДАНИХ В МОДУЛЬ КРЕЙТА:

1) записати біти W24 - W17 в регістр 0;

2) записати біти W16 - W9 в регістр 1;

3) записати біти W8 - W1 в регістр 2;

4) записати субадресу A в регістр 3;

5) записати функцію F в регістр 4;

6) записати номер станції N в регістр 5;

7) ініціювати цикл КАМАК шляхом запису будь-яких даних в регістр 7;

8) перевірити результат виконання циклу КАМАК (біти X і Q), прочитавши вміст регістру 8.

ОПЕРАЦІЯ ЧИТАННЯ ДАНИХ З КРЕЙТУ:

1) записати субадресу A в регістр 3;

2) записати функцію F в регістр 4;

3) записати номер станції N в регістр 5;

4) ініціювати цикл КАМАК шляхом запису будь-яких даних в регістр 7;

5) перевірити результат виконання циклу КАМАК (біти X і Q), прочитавши вміст регістру 8;

6) прочитати біти R24 - R17 з регістру 9;

7) прочитати біти R16 - R9 з регістру A;

8) прочитати біти R8 - R1 з регістру B;

Адресація регістрів контролера ФК4410 з ЕОМ здійснюється через інтерфейсну плату. Сигнали на адресних лініях A9 - A4 системної шини ЕОМ типу PC XT/AT визначають базову адресу плати, яка задана числом 3A0H, а чотири молодших лінії A3 - A0 використовуються для адресації регістрів контролера і регістра керування інтерфейсної плати. Тобто, для адресації регістрів контролера з ЕОМ потрібно до базової адреси 3A0H додати відносну адресу (див. табл. 7.2) вибраного регістру.

Адреса 3AFH відноситься до керуючого 8-ми бітового регістру інтерфейсної плати, який програмується наступним чином. В розряди D0 - D6 цього регістру можна лише писати, а з розряду D7 інформація лише читається. Розряди D0 - D3, D6, D7 використовуються для програмування і контролю (розряд D7) роботи КАМАК в режимі прямого доступу до пам'яті. Розряди D5, D4 служать для програмування номера крейта, з яким ЕОМ обмінюється інформацією. Інтерфейсна плата дозволяє працювати з двома крейтами. Значення D5=0, D4=0 встановлюють обмін інформацією з крейтом, який приєднаний до з'єднувача "0", а комбінація D5=0, D4=1 активізує крейт, приєднаний до з'єднувача "1".

Система команд модуля цифро-аналогового перетворення ФК70, який використовується в даній роботі. Модуль ФК70 служить для перетворення чотирнадцяти розрядного двійкового коду з додатковим знаковим розрядом в постійну напругу додатної або від'ємної полярності. Модуль містить два однакових цифро-аналогових

перетворювача ЦАП1 і ЦАП2. Він виготовлений в стандартному корпусі і займає одну станцію крейта КАМАК. На передній панелі є засоби індикації запиту на обслуговування (світлові діоди) та встановлені з'єднувачі: виходи ЦАП1 і ЦАП2 з написом OUT і входи лічильників-регістрів ЦАП1 і ЦАП2 з написом COUNT, а також спільний для обох ЦАП вхід керування напрямком рахунку лічильників з позначкою

"+⌊-" (високий потенціал на цьому вході встановлює режим рахунку на додавання, а низький - на віднімання)

Позначення адресних команд складається з позначень сигналів N, A, F. В круглих дужках після літер A і F вказуються числа, які відповідають субадресі і номеру функції. Конкретне значення N не вказується, оскільки воно позначає номер станції крейту на якій встановлений модуль і залежить від конфігурації експерименту.

N A(0) F(16) - запис коду в регістр ЦАП1. При виконані цієї команди дані з ліній запису W магістралі крейту передаються в регістр цифро-аналогового перетворювача. Після виконання команди через 10 мкс (час перетворення) на виході OUT1 встановлюється аналоговий сигнал.

N A(1) F(16) - запис коду в регістр ЦАП2. При виконані цієї команди дані з ліній запису W магістралі крейту передаються в регістр цифро-аналогового перетворювача. Після виконання команди через 10 мкс (час перетворення) на виході OUT2 встановлюється аналоговий сигнал.

Двійковий код даних поступає по лініях W1 - W14, W16 і має формат, який показаний на мал. 7.5.

N A(3) F(26) - дозвіл рахунку. Ця команда дозволяє проходження сигналів з входів COUNT (для обох ЦАП) на входи регістрів-лічильників ЦАП, які можуть працювати в режимах: рахунок на додавання; рахунок на віднімання.

N A(3) F(24) - заборона рахунку. Ця команда забороняє (для обох ЦАП) проходження сигналів з входів COUNT1 і COUNT2 на входи регістрів-лічильників ЦАП.

N A(2) F(26) - дозвіл запиту. Коли регістри-лічильники ЦАП працюють в режимі рахунку і має місце переповнення регістру (лічильник працює в режимі рахунок на додавання) або в усіх розрядах регістру втановлюються нулі (лічильник працює в режимі рахунок на віднімання), то виробляється сигнал кінця рахунку. Цей сигнал передається на вихід модуля, як сигнал L запиту на обслуговування.

N A(2) F(24) - заборона запиту. Ця команда забороняє модулю виставляти сигнал L.

N A(2) F(8) - перевірка запиту модуля. При виконані цієї команди виробляється відповідь модуля Q=1, якщо в модулі один з лічильників-регістрів ЦАП (або обидва) виробив сигнал кінця рахунку.

N A(0) F(8) - перевірка запиту 1. При виконані цієї команди модуль виробляє відповідь Q=1, якщо лічильник-регістр ЦАП1 виробив сигнал кінця рахунку.

N A(0) F(8) - перевірка запиту 2. При виконанні цієї команди модуль виробляє відповідь Q=1, якщо лічильник-регістр ЦАП2 виробив сигнал кінця рахунку.

N A(0) F(10) - очищення запиту 1. В результаті виконання цієї команди тригер, який формує сигнал L запиту від ЦАП1, очищується, тобто на його виході встановлюється лог. 0.

N A(1) F(10) - очищення запиту 2. В результаті виконання цієї команди тригер, який формує сигнал L запиту від ЦАП2, очищується, тобто на його виході встановлюється лог. 0.

Безадресні команди Z, C, S1, S2. Ці команди виконуються коли на відповідних лініях магістралі крейту виробляються сигнали:

Z - початкове встановлення. По цій команді відбувається заборона рахунку лічильників, заборона запиту, очищення запиту 1 і очищення запиту 2. Дії при виконанні цієї команди еквівалентні виконанню команд: *N A(3) F(24), N A(2) F(24), N A(0) F(10), N A(1) F(10).*

C - очищати. По цій команді очищуються регістри ЦАП1 і ЦАП2 (регістри першої групи).

S1, S2 - ці команди формують на лініях магістралі крейту відповідні сигнали стробування.

Завдання для домашньої підготовки

1. Ознайомтесь з основними стандартами системи модульної електроніки КАМАК.

2. Ознайомтесь з операціями КАМАК (системою команд КАМАК).

3. Ознайомтесь з логічною структурою контролера крейта ФК4410, який використовується в даній роботі. Вивчіть способи адресації регістрів цього контролера.

4. Вивчіть систему команд модуля цифро-аналогового перетворення ФК70, який використовується в даній роботі.

5. Напишіть варіанти програм для виконання команд КАМАК, які використовуються в модулі ФК70.

Програми для виконання команд КАМАК можуть бути написані на мові високого рівня, використовуючи функції і оператори цих мов, які працюють з портами, або на мові Асемблера. Приклад програми на мові BASIC (програма 7.1), яка реалізує ініціювання крейту і записує шістнадцяти розрядний двійковий код 0000111111110000B в регістр ЦАП1 модуля ФК70, який знаходиться на 10-й станції крейта (адресна команда *N(10)A(0)F(16)*).

Програма 7.1.

```
10 P=&H3A0
20 OUT P+15,0
30 OUT P+1,15
40 OUT P+2,240
50 OUT P+3,0
60 OUT P+4,16
70 OUT P+5,10
80 OUT P+7,0
90 D=INP(P+8)
100 PRINT D AND 3
110 END
```

Програма 7.2 реалізує безадресну команду очищення регістрів ЦАП1 і ЦАП2 (генерує сигнал С на лінії магістралі крейту).

Програма 7.2.

```
10 P=&H3A0
20 OUT P+6,2
30 OUT P+7,0
40 END
```

Завдання до лабораторної роботи

Завдання 1. Ознайомлення з конструкцією крейту КАМАК, функціональних модулів та порядком вмикання живлення крейту.

Завдання 2. Виконати команду *Початкове встановлення*. Проконтролювати її виконання по станах світлових діодів модуля індикатора магістралі. Прочитати вміст регістру 8 (див. табл. 7.2). Записати та пояснити значення розрядів цього регістру.

Завдання 3. Виконати команди модуля ФК70: *Заборона запиту* та *Дозвіл запиту*. Проконтролювати їх виконання по станах світлових діодів модуля індикатора магістралі. Прочитати вміст регістру 8 (див. табл. 7.2) після кожної команди. Записати та пояснити значення розрядів цього регістру.

Завдання 4. Виконати команди модуля ФК70: *Запис коду в регістр ЦАП1* та *Запис коду в регістр ЦАП2*. Порядок виконання завдання:

- до виходу цифро-аналогового перетворювача “OUT1” приєднати вхід вольтметра ВК7-21;

- в регістр ЦАП1 записати код 0011111010000000B;

- проконтролювати по вольтметру, що напруга на виході ЦАП1 установилась +10 В;

- в регістр ЦАП1 записати код 1011111010000000B;

- проконтролювати по вольтметру, що напруга на виході ЦАП1 установилась -10 В;

- користуючись таблицею 2 технічного опису модуля ФК70 (Блок цифро-аналоговых преобразователей ФК70. Техническое описание и инструкция по эксплуатации. ЗПИ.499.354 ТО) сформувати двійковий код, який потрібно записати в регістр ЦАП1, щоб на його виході установилась напруга +5 В (-5В).

Завдання 5. Формування змінної в часі напруги на виході ЦАП2. Порядок виконання завдання:

- приєднати до входу “COUNT2” генератор імпульсів;

- приєднати до виходу ЦАП2 “OUT2” вхід вольтметра ВК7-21;

- встановити частоту слідування імпульсів в межах 200 - 500 Гц;

- на вхід керуванням напрямку рахунку подати додатну напругу в межах 3 - 5 В;

- виконати керуючі команди модуля ФК70 в наступній послідовності: очищення регістрів ЦАП1 і ЦАП2; дозвіл запиту; дозвіл рахунку.

- спостерігати по вольтметру зміну в часі напруги на виході “OUT2”;

- діждатися кінця рахунку лічильника (коли засвітиться індикатор “Запит 2”);

- виконати керуючі команди модуля ФК70 в наступній послідовності: перевірка запиту; перевірка запиту 1; перевірка запиту 2; очищення запиту. Після кожної команди прочитати вміст регістру 8 (див. табл. 7.2). Записати та пояснити значення розрядів цього регістру.

Зміст звіту

Звіт повинен містити:

- назву та мету роботи;

- структурну схему і опис системи КАМАК та її зв’язку з ЕОМ;

- результати завдань 2 - 5 та програми, які використовувались для обміну інформацією між контролером крейту і ЕОМ при виконанні цих завдань.

Додаткові завдання

Д7.1. Напишіть програму, яка формує на виході ЦАП1 змінну в часі напругу виду $U=Uo\cdot\sin\omega t$, де Uo=5 В, а $\omega=2\pi/10$ с-1.

Д7.2. Напишіть програму, яка формує на виході ЦАП2 змінну в часі напругу виду $U=k\cdot t$, де t - в секундах, k=1 В/с.

Д7.3. Намалюйте структурну схему системи КАМАК для вимірювання ВАХ напівпровідникових діодів. Напишіть алгоритм керуючої програми, яка реалізує ці вимірювання.

Д7.4. Напишіть програму, яка виводить результати вимірювання ВАХ діода на двокоординатний електронний потенціометр у вигляді графіка.

Контрольні запитання

1. Що називають системою модульної електроніки КАМАК, назвіть її основні компоненти і їх призначення?
2. Що є спільним в організації системи КАМАК і мікро-ЕОМ.
3. Яке основне призначення крейта КАМАК?
4. Назвіть основні стандарти системи КАМАК та поясніть їх призначення.
5. Що називають магістраллю крейта? Поясніть її структуру.
6. Яке призначення мають команди (операції) КАМАК?
7. Назвіть та поясніть типи сигналів на магістралі крейта.
8. Яка різниця між командами керування Z і C?
9. Яке призначення сигналу X? Який модуль крейта може його виробляти?
10. Які функції виконує сигнал запиту L функціонального модуля?
11. Для чого в стандарті КАМАК використовується два сигнали стробування S1 і S2?
12. Які модулі системи КАМАК можуть ініціювати обмін інформацією між ЕОМ і функціональним модулем?
13. Поясніть, як здійснюється обмін інформацією по магістралі крейта.
14. Поясніть запис N(7)A(3)F(16).
15. Поясніть логічну структуру контролера крейту. Яке призначення його регістрів?

16. В якій послідовності програмуються регістри контролера крейту ФК4410 при записувані даних в функціональний модуль?

Розділ перший

ЗАГАЛЬНІ ВІДОМОСТІ

1. Навчальна мікро-ЕОМ

Навчальна мікро-ЕОМ (навчальний мікропроцесорний комплект "УМК") призначена для вивчення основ проектування мікро-ЕОМ на мікропроцесорних великих інтегральних схемах (ВІС) з фіксованим набором команд і може бути використана для дослідження методів програмування та роботи ВІС, які входять в мікропроцесорний комплект серії КР580. Мікро-ЕОМ типу УМК може виконувати функцію керуючої ЕОМ при створені та досліджені роботи систем керування різними об’єктами. Вона являється зручним мікропроцесорним засобом для відлагодження порівняно невеликих (до 0,5 К байт) прикладних програм. Наявність великої кількості засобів індикації дозволяє наглядно досліджувати процеси перетворення та передачі інформації в мікро-ЕОМ.

1.1. Основні функції мікро-ЕОМ

Уведення інформації в мікро-ЕОМ та вибір директив здійснюється з клавіатури, яка розміщена на панелі керування та індикації. Відображення інформації, яка уводиться в мікро-ЕОМ і виводиться з неї, здійснюється в шістнадцятковому коді на шестирозрядному дисплеї.

З клавіатури викликаються наступні директиви:

- читання та модифікація вмісту комірок пам'яті;
- читання та модифікація вмісту регістрів мікропроцесора;
- підрахунок контрольної суми масиву пам'яті;
- заповнення масиву пам’яті константою;
- переміщення заданого масиву пам'яті в адресному просторі;
- виконання прикладних програм з можливістю уведення двох точок зупинки програми.

1.2. Структура навчальної мікро-ЕОМ

На мал. 0.1 приведена структура мікро-ЕОМ де вказані окремі функціональні блоки та вузли. Мікропроцесор КР580ВМ80А виконує роль центрального процесорного пристрою. Обмеженість енергії, яка може бути розсіяна на кристалі, не дозволяє забезпечити достатню навантажувальну здатність виходів мікропроцесора (МП). Тому, в більшості випадків, потрібно включати додаткові мікросхеми (МС) в ролі зовнішніх буферних схем. Зовнішня магістраль в даній ЕОМ представлена трьома окремими шинами: шиною даних (ШД), шиною адреси (ША) та шиною керування (ШУ). Відповідно використовуються буферні схеми: буфер шини даних (БШД) та буфер адресної шини (БША).

Конструктивно шини виконані у вигляді наборів провідників, які з'єднують виводи МП і МС та контакти з'єднувачів, в які вставляються модулі мікро-ЕОМ. МП серії КР580 оперує з восьмирозрядними словами даних, тому ШД складається з восьми окремих провідників, по яких передається інформація у вигляді двійкового коду. Рівні сигналів відповідають стандартним рівням ТТЛ-схем. Шина даних - двонапрямна. Інформація по ній може передаватись як в операційний пристрій, так із нього. Напрям передачі встановлює МП.

Шина керування теж складається із набору окремих провідників по кожному з яких передаються певні сигнали в певному напрямку. МП серії КР580 виготовлений в корпусі, який має 40 виводів (мал. 1.1). Цих виводів недостатньо, щоб кожному сигналові керування надати окремий провідник. Тому частина сигналів керування передається по шині даних в режимі розділення часу. Цим пояснюється наявність схеми формування сигналів керування (СФСК).

Розрядність шини адреси визначає максимальний об'єм пам'яті мікро-ЕОМ, до якої може безпосередньо адресуватись МП. Чим більший об'єм пам'яті, тим більш складні програми може виконувати ЕОМ і тим потужніші її обчислювальні та керуючі можливості. МП серії КР580 має 16-розрядну адресну шину. Це означає, що операційний пристрій може звертатись до 216=65536 комірок пам'яті. Враховуючи, що в одній комірці зберігається 8 біт (1 байт), одержуємо, що максимальний об'єм пам'яті в даному випадку 64 К байт (1 К байт=1024 байт). Шина адреси - однонапрямна. Код адреси поступає на постійний запам'ятовуючий пристрій (ПЗПр), оперативний запам'ятовуючий пристрій (ОЗПр) та на схему дешифратора адреси (СДА).

Пульт оператора складається: із клавіатури, для уведення шістнадцяткових кодів чисел; шістирозрядного дисплея, чотири розряди якого служать для відображення коду адреси в шістнадцятковій системі числення, а два - для відображення даних; світлових діодів, для світлової індикації станів ШД, ША і ШК; кнопок керування - перезапуску мікро-ЕОМ - "СБ"; переривання роботи (виконання програми) - "ПР"; вибору по-крокового режиму роботи: "РБ/ШГ" - по-кроковий режим (кнопка натиснута), "КМ/ЦК" - команда/цикл (команди програми виконуються по машинних циклах коли ця кнопка натиснута), "ШГ" - зробити наступний крок.

Основою мікро-ЕОМ є мікропроцесор, який виконує операції по обробці інформації. Вихідним станом мікропроцесора є читання інформації по нульовій адресі з ПЗПр, в який він переходить після того, як була натиснута кнопка "СБ".

В ПЗПр (мікросхема К573РФ2) записана програма "Монітор", яка забезпечує уведення інформації з клавіатури пульта оператора, виведення даних і адрес на дисплей та виконання директив. Програма “Монітор” займає 1 К байт пам’яті ПЗПр і використовує 54 комірки ОЗПр.

ОЗПр використовується для зберігання досліджуваних і прикладних програм. Ємність ОЗПр 1 К байт (дві мікросхеми К541РУ2).

Схема керування кроковим режимом (СККР) переводить ОП в стан ОЧІКУВАТИ після виконання наступного кроку. Можливі два режими роботи: по-командний та по-цикловий. Кроковий режим встановлюється перемикачем "РБ/ШГ", вибір величини кроку - перемикачем "КМ/ЦК". Для виконання кроку потрібно натиснути кнопку "ШГ", при цьому, після виконання кроку програми, на світловій індикації відображається стан адресної шини, шини даних та регістру стану процесора (РСП) у двійковому коді (лог. "1" відповідає свічення діодів).

Виконання досліджуваної програми може бути зупинене натиснувши на кнопку "ПР". При цьому вміст всіх регістрів МП зберігається в ОЗПр.

1.3. Адресація в навчальній мікро-ЕОМ

Таблиця 0.1. Структура адресного простору мікро-ЕОМ

Адреса (Н-коди)Вміст пам’ятіТип пам’ятіОб’єм0000 03FF 0400 07FFКеруюча програма “Монітор” Демонстраційні програми ПЗПр 2 кб0800 0AFF 0B00 0BAF 0BDA 0BFFОбласть для запису досліджуваних програм Область для запису даних досліджуваних програм Вершина стеку програми “Монітор” ОЗПр 1 кб

На мал. 0.2 приведена карта пам’яті мікро-ЕОМ УМК. З якої видно, що перші 2 К байт адрес відносяться до ПЗПр в якому записані управляюча програма “Монітор” і демонстраційні програми. Адреси з 0800H до 0BFFH відносяться до оперативної пам’яті. Адреса 0BDAH - вершина стекової пам’яті. Комірки пам’яті з адреси 0BDBH по 0BFFH використовуються керуючою програмою для запису даних. Адреса 0800H є початковою адресою оперативної пам’яті. Більшість із приведених в лабораторному практикумі програм розпочинаються з адреси 0800H. Для запису даних при виконанні програм необхідно використовувати область пам’яті з адреси 0B00H до 0BB0H.

1.4. Режими роботи навчальної мікро-ЕОМ

Керуюча програма мікро-ЕОМ представляє собою діалогову систему і служить для реалізації функцій, перерахованих в п. 1.1. Для виконання потрібної функції оператор повинен увести з клавіатури відповідну директиву та необхідні параметри. Результат виконання команди буде відображений на дисплеї.

Дисплей складається із шести семисегментних світлодіодних матриць, які розміщені на передній панелі УМК. Світлодіодні матриці використовуються для відображення даних типу "АДРЕСА" (ліві матриці) та типу "ДАНІ" (дві праві матриці).

Клавіатура також поділена на дві частини: функціональну - лівий набір кнопок для уведення директив і числову - правий набір кнопок для уведення параметрів. Директивні клавіші мають наступні позначення:

"П" - читання та модифікація вмісту пам’яті;

"РГ" - читання та модифікація вмісту регістрів МП;

"СТ" - запуск прикладної програми на виконання;

"КС" - підрахунок контрольної суми;

"ЗК" - заповнення масиву пам’яті константою;

"ПМ" - копіювання областей пам’яті;

“|__|“ - "пропуск", для розділення параметрів при їх уведенні;

"ВП" - "виконати", означає кінець директиви.

Числові кнопки служать для уведення чисел в шістнадцятковому коді. Кнопки з 4 по F служать ідентифікаторами регістрів мікропроцесора.

При неправильній роботі з клавіатурою в крайній правій позиції дисплея висвічується знак "?".

Робота мікро-ЕОМ запрограмована так, що після вмикання живлення шляхом натиснення кнопки із значком “~“ і натиснення кнопки "СБ" запускається програма Монітор. Ця програма починається з адреси 0000Н. В результаті відбувається завантаження ОЗПр вихідними значеннями регістрів, гасіння індикаторів і виведення на дисплей (зліва) символу "–", який свідчить про готовність мікро-ЕОМ до роботи. Після цього ЕОМ переходить в режим приймання директив оператора. В загальному вигляді кожну директиву можна представити у вигляді:

КОП [ПАР1, ПАР2, ПАР3] ВП,

де КОП - ідентифікатор директиви, який відповідає одній із функціональних кнопок; ПАР1, ПАР2, ПАР3 - параметри директив, може бути від одного до трьох параметрів, які розділяються кнопкою “пропуск”; ВП - кнопка, натиснення якої ініціалізує виконання директиви.

Для зупинки виконання досліджуваної програми використовується кнопка "ПР". При цьому керування передається підпрограмі обробки переривання командою RST7. Ця підпрограма зберігає в ОЗПр вміст всіх регістрів мікропроцесора і передає керування Монітору. На дисплеї відображається вміст лічильника команд. Це число в Н-кодах на одиницю більше адреси останнього байта останньої виконаної команди. Виконання перерваної програми може бути продовжене починаючи з адреси, на якій зупинилась програма, або з будь-якої іншої адреси.

Для успішного виконання лабораторних робіт необхідно попередньо ознайомитися з системою команд мікропроцесора КР580ВМ80А, або мікропроцесорів І8080, І8085 фірми Intel, Z80 фірми Zilog та з програмуванням цих МП в машинних кодах і на мові Асемблера.

2. Методика проведення лабораторних занять

В процесі лабораторних занять реалізується один із самих важливих моментів навчального процесу - зв'язок теорії з практикою, в результаті чого студент отримує необхідні знання та практичні навики у проектуванні мікро-ЕОМ на базі мікропроцесорних комплектів інтегральних схем та у розробці алгоритмів і написанні програм на мові Асемблера. Кожне лабораторне заняття передбачає досягнення певних виховної і навчальної мети.

Виховна мета досягається цілеспрямованим впливом викладача і повинна сприяти прагненню студентів глибоко засвоїти суть виконуваних ними робіт, оволодіти практичними навиками роботи з мікропроцесорними пристроями.

Навчальну мету заняття можна звести до наступних основних моментів:

- закріплення студентами теоретичних положень дисципліни, яка вивчається;

- вивчення логічної структури мікропроцесорних пристроїв, системи команд типових мікропроцесорів та принципів обробки інформації в них;

- вивчення методів написання програм та їх налагоджування;

Виходячи із мети лабораторних занять, сучасних задач, які ставляться перед ними, повинна вибиратися методика підготовки та проведення лабораторних занять.

В основу виконання лабораторних робіт покладений фронтальний метод, згідно якого експериментальні дослідження проводяться після того, коли матеріал даної теми викладений на лекції і студенти, на лабораторному занятті, виконують однакові завдання і працюють з однаковими апаратними засобами. Така організація навчальних занять сприяє закріпленню і розширенню знань студентів. Фронтальне проведення лабораторних робіт дозволяє викладачу одночасно керувати і слідкувати за роботою студентів всієї групи, давати для всієї групи вказівки в процесі виконання лабораторної роботи, використовуючи технічні засоби навчання, і пояснювати характерні помилки студентів, які допущені в процесі виконання роботи.

Виконання кожної лабораторної роботи складається із двох етапів.

1. Підготовка до лабораторної роботи, вивчення теоретичного матеріалу, ознайомлення з структурою та алгоритмом досліджуваних програм.

2. Ознайомлення з апаратними засобами, які використовуються на занятті (мікро-ЕОМ, пристрої уведення-виведення, пристрої індикації та ін.), а також підготовка, уведення в навчальну ЕОМ досліджуваних програм, їх налагодження та виконання.

Порядок виконання досліджень в лабораторії.

1. Студент допускається до виконання наступної роботи при наявності частково підготовленого звіту по ній, згідно вказаних вище вимог.

2. Після дозволу виконувати дослідження студент вмикає живлення навчальної мікро-ЕОМ, завантажує досліджувані програми та виконує завдання, які вказані в пункті "Завдання до лабораторної роботи".

3. Заліком виконаної роботи є демонстрація студентом викладачу роботи відлагодженного пристрою, або результату виконання на мікро-ЕОМ досліджуваної (розробленої) програми.

4. До наступної лабораторної роботи повністю оформляється звіт за виконану роботу, який підписується викладачем на наступному занятті після занесення в нього всіх завдань, перерахованих в пункті "Зміст звіту".

Перед виконанням кожної лабораторної роботи викладач опитує студентів як по змісту самої роботи, так і по методиці її виконання. Непідготовленні студенти не допускаються до виконання лабораторної роботи і вивчають в лабораторії незасвоєний ними матеріал по рекомендованій літературі.

До кожної лабораторної роботи є перелік додаткових завдань, які орієнтовані на індивідуальну роботу студентів. Після виконання обов'язкових завдань студент може виконувати ці завдання. Результати виконання додаткових завдань оцінюються відповідними оцінками, які враховуються при складані заліків (екзаменів) по цій дисципліні.

Описи лабораторних робіт складені настільки детально, містять вказівки, що і як робити, лістинг досліджуваних програм з коментарями, щоб дати можливість студентам самостійно (без підказування викладача та участі персоналу лабораторії) виконувати обов`язкові завдання.

3. Оформлення звіту за виконану роботу

Звіти за виконані роботи оформляються в окремому зошиті. При підготовці до лабораторної роботи обов'язковим є часткове оформлення звіту по цій роботі. Він повинен містити назву та мету роботи, самостійно розроблені програми, які вказані в пункті "Завдання для домашньої підготовки".

Після виконання лабораторної роботи у звіт заносяться схеми, програми та інша інформація згідно вказівок, які містяться в пункті "Зміст звіту". Додаткові завдання, якщо вони виконувались, заносяться у звіт з детальним описом методики виконання та аналізом отриманих результатів.

Звіти оформляються згідно вимог стандарту до текстових документів. Малюнки, креслення структурних та принципових схем виконуються олівцем з дотримання стандарту на умовні графічні зображення елементів схем.

До наступної лабораторної роботи повністю оформляється звіт за попередню виконану роботу. Студенти, які не оформили звіт, до виконання наступної роботи не допускаються і оформляють його в лабораторії.

Звіти за виконані роботи зберігаються у студентів.

СПИСОК ЛІТЕРАТУРИ

Додаток 1.

Система команд мікропроцесора I8080 фірми Intel

Система команд мікропроцесора I8080 містить одно- дво- і трьохбайтові команди. Першим байтом команди завжди є код операції (КОП), який визначає характер дій, які виконує мікропроцесор. Набір ыэЬэыбыэбыЬэыЪыэЪыЪыЪы

ж коди операцій.

Кожна команда ініціалізує виконання певних дій мікропроцесором. Всі команди можна поділити на п’ять груп: передачі даних; арифметичні; логічні; переходів; керування, уведення-виведення та роботи із стеком.

Команди передачі даних використовуються для передачі даних між регістрами, або пам’яттю і регістрами.

Арифметичні команди виконують операції додавання, віднімання, інкременту і декременту над даними, які розміщені в регістрах і в пам’яті.

Логічні команди виконують операції І, АБО, АБО З ВИКЛЮЧЕННЯМ, інвертування, порівняння над даними, які розміщені в регістрах і в пам’яті.

Команди переходів використовуються для організації переходів по умові і без умови та переходу на підпрограми і повернення з них.

Команди керування, уведення-виведення та роботи зі стеком використовуються для керування перериваннями, встановлення бітів регістру ознак та уведення і виведення інформації.

В таблиці Д1.1 приведений склад команд мікропроцесора I8080 в алфавітному порядку. Коди операцій записані в шістнадцятковій системі числення.

Таблиця Д1.1. Склад команд мікропроцесора Intel 8080

МНЕМОНІКАКОПОписADD A ADD B ADD C ADD D ADD E ADD H ADD L87 80 81 82 83 84 85Додати A до A (подвоєння A) Додати B до A Додати C до A Додати D до A Додати E до A Додати H до A Додати L до AADD M86Додати вміст пам’яті LOC (HL) до AADI υC6Додати безпосередньо наступні дані υ до AADC A ADC B ADC C ADC D ADC E ADC H ADC L8F 88 89 8A 8B 8C 8DДодати A до A та біт перенесення (подвоєння A) Додати B до A та біт перенесення Додати C до A та біт перенесення Додати D до A та біт перенесення Додати E до A та біт перенесення Додати H до A та біт перенесення Додати L до A та біт перенесенняADC M8EДодати пам’ять LOC (HL) до A та біт

перенесенняACI υCEДодати безпосередньо наступні дані υ до A та біт перенесенняANA A ANA B ANA C ANA D ANA E ANA H ANA LA7 A0 A1 A2 A3 A4 A5Тест A A ∧ A Логічна операція I B ∧ A Логічна операція I C ∧ A Логічна операція I D ∧ A Логічна операція I E ∧ A Логічна операція I H ∧ A Логічна операція I L ∧ AANA MA6Логічна операція I пам'ять LOC (HL) ∧ AANI υE6Безпосередньо наступні дані υ ∧ ACALL *aa*CDПерейти на підпрограму за адресою *aa*CZ *aa* CNZ *aa* CP *aa* CM *aa* CC *aa* CNC *aa* CPE *aa* CPO *aa*CC C4 F4 FC DD D4 EC E4Перейти на підпрограму за адресою *aa*, якщо ноль Перейти на підпрограму за адресою *aa*, якщо не ноль Перейти на підпрограму за адресою *aa*, якщо плюс Перейти на підпрограму за адресою *aa*, якщо мінус Перейти на підпрограму за адресою *aa*, якщо є перенесення Перейти на підпрограму за адресою *aa*, якщо немає перенесення Перейти на підпрограму за адресою *aa*, якщо парно Перейти на підпрограму за адресою *aa*, якщо непарноCMA CMC2F 3FІнвертувати A Інвертувати біт перенесенняCMP A CMP B CMP C CMP D CMP E CMP H CMP LBF B8 B9 BA BB BC BDВстановити індикатор ноля операцією (A)-(A) Порівняти A з B Порівняти A з C Порівняти A з D Порівняти A з E Порівняти A з H Порівняти A з LCMP MBEПорівняти A з вмістом пам'яті LOC (HL)CPI υFCПорівняти A з безпосередньо наступними даними υDAA27Десяткова корекція акумуляторaDAD B DAD D DAD H DAD SP09 19 29 39Додати BC з HL Додати DE з HL Додати HL з HL (подвоєння HL) Додати SP з HLDCR A DCR B DCR C DCR D DCR E DCR H DCR L3D 05 0D 15 1D 25 2DДекрементувати A Декрементувати B Декрементувати C Декрементувати D Декрементувати E Декрементувати H Декрементувати LDCR M35Декрементувати вміст пам'яті LOC (HL)DCX B DCX D DCX H DCX SP0B 1B 2B 3BДекрементувати BC Декрементувати DE Декрементувати HL Декрементувати SPDIF3Заборонити перериванняEIFBДозволити перериванняHLT76Зупинити мікропроцесорIN υDBУвести дані з пристрою υINR A INR B INR C INR D INR E INR H INR L3C 04 0C 14 1C 24 2CІнкрементувати A Інкрементувати B Інкрементувати C Інкрементувати D Інкрементувати E Інкрементувати H Інкрементувати LINR M34Інкрементувати вміст пам'яті LOC (HL)INX B INX D INX H INX SP03 13 23 33Інкрементувати BC Інкрементувати DE Інкрементувати HL Інкрементувати SPJMP *aa*C3Перейти за адресою *aa*JZ *aa* JNZ *aa* JP *aa* JM *aa* JC *aa* JNC *aa* JPE *aa* JPO *aa*CA C2 F2 FA DA D2 EA E2Перейти за адресою *aa*, якщо ноль Перейти за адресою *aa*, якщо не ноль Перейти за адресою *aa*, якщо плюс Перейти за адресою *aa*, якщо мінус Перейти за адресою *aa*, якщо є перенесення Перейти за адресою *aa*, якщо немає перенесення Перейти за адресою *aa*, якщо парітет парний Перейти за адресою *aa*, якщо парітет

непарнийLDA *aa*3AЗавантажити A із джерела з адресою *aa*LDAX B LDAX D0A 1AЗавантажити A з комірки пам’яті LOC (BC) Завантажити A з комірки пам’яті LOC (DE)LHLD *aa*2AЗавантажити HL із джерела з адресою *aa*LXI B, υυ01Завантажити BC безпосередньо наступними даними υυLXI H, υυ LXI SP, υυ21 31Завантажити HL безпосередньо наступними даними υυ Завантажити SP безпосередньо наступними даними υυMOV A, B MOV A, C MOV A, D MOV A, E MOV A, H MOV A, L78 79 7A 7B 7C 7DПередати дані з B в A Передати дані з C в A Передати дані з D в A Передати дані з E в A Передати дані з H в A Передати дані з L в AMOV A, M7EПередати дані з комірки пам’яті LOC (HL) в AMOV B, A MOV B, C MOV B, D MOV B, E MOV B, H MOV B, L47 41 42 43 44 45Передати дані з A в B Передати дані з C в B Передати дані з D в B Передати дані з E в B Передати дані з H в B Передати дані з L в BMOV B, M46Передати дані з комірки пам’яті LOC (HL) в BMOV C, A MOV C, B MOV C, D MOV C, E MOV C, H MOV C, L4F 48 4A 4B 4C 4DПередати дані з A в C Передати дані з B в C Передати дані з D в C Передати дані з E в C Передати дані з H в C Передати дані з L в CMOV C, M4EПередати дані з комірки пам’яті LOC (HL) в CMOV D, A MOV D, B MOV D, C MOV D, E MOV D, H MOV D, L57 50 51 53 54 55Передати дані з A в D Передати дані з B в D Передати дані з C в D Передати дані з E в D Передати дані з H в D Передати дані з L в DMOV D, M56Передати дані з комірки пам’яті LOC (HL) в DMOV E, A MOV E, B5F 58Передати дані з A в E Передати дані з B в EMOV E, C MOV E, D MOV E, H MOV E, L59 5A 5C 5DПередати дані з C в E Передати дані з D в E Передати дані з H в E Передати дані з L в EMOV E, M5EПередати дані з комірки пам’яті LOC (HL) в EMOV H, A MOV H, B MOV H, C MOV H, D MOV H, E MOV H, L67 60 61 62 63 65Передати дані з A в H Передати дані з B в H Передати дані з C в H Передати дані з D в H Передати дані з E в H Передати дані з L в HMOV H, M66Передати дані з комірки пам’яті LOC (HL) в HMOV L, A MOV L, B MOV L, C MOV L, D MOV L, E MOV L, H6F 68 69 6A 6B 6CПередати дані з A в L Передати дані з B в L Передати дані з C в L Передати дані з D в L Передати дані з E в L Передати дані з H в LMOV L, M6EПередати дані з комірки пам’яті LOC (HL) в LMOV M, A MOV M, B MOV M, C MOV M, D MOV M, E MOV M, H MOV M, L77 70 71 72 73 74 75Передати дані в комірку пам’яті LOC (HL) з A Передати дані в комірку пам’яті LOC (HL) з B Передати дані в комірку пам’яті LOC (HL) з C Передати дані в комірку пам’яті LOC (HL) з D Передати дані в комірку пам’яті LOC (HL) з E Передати дані в комірку пам’яті LOC (HL) з H Передати дані в комірку пам’яті LOC (HL) з LMVI A, υ MVI B, υ MVI C, υ MVI D, υ MVI E, υ MVI H, υ MVI L, υ3E 06 0E 16 1E 26 2EПередати безпосередньо

наступні дані υ в A Передати безпосередньо наступні дані υ в B Передати безпосередньо наступні дані υ в C Передати безпосередньо наступні дані υ в D Передати безпосередньо наступні дані υ в E Передати безпосередньо наступні дані υ в H Передати безпосередньо наступні дані υ в LMVI M, υ 36Передати безпосередньо наступні дані υ в LOC (HL)NOP00Нема операційORA A ORA B ORA C ORA D ORA E ORA H ORA LB7 B0 B1 B2 B3 B4 B5Перевірити A і скинути перенос Логічна операція АБО B ∨ A Логічна операція АБО C ∨ A Логічна операція АБО D ∨ A Логічна операція АБО E ∨ A Логічна операція АБО H ∨ A Логічна операція АБО L ∨ AORA MB6Вміст комірки пам'яті LOC (HL) ∨ AORI υF6Безпосередньо наступні дані υ ∨ AOUT υD3Вивести вміст A за адресою υPCHLE9Передати вміст H і L в PC (лічильник команд)POP B POP D POP H POP PSWC1 D1 E1 F1Вилучити із стеку вміст пари регістрів BC Вилучити із стеку вміст пари регістрів DE Вилучити із стеку вміст пари регістрів HL Вилучити із стеку слово стану процесора PSWPUSH B PUSH D PUSH H PUSH PSWC5 D5 E5 F5Завантажити в стек вміст пари регістрів BC Завантажити в стек вміст пари регістрів DE Завантажити в стек вміст пари регістрів HL Завантажити в стек слово стану процесора PSWRAL RAR RLC RRC17 1F 07 0FПеремістити циклічно CY+A ліворуч Перемістити циклічно CY+A праворуч Перемістити A ліворуч з перенесенням Перемістити A праворуч з перенесеннямRETC9Повернення з підпрограмиRZ RNZ RP RM RC RNC RPE RPOC8 C0 F0 F8 D8 D0 E8 E0Повернення з підпрограми, якщо ноль Повернення з підпрограми, якщо не ноль Повернення з підпрограми, якщо плюс Повернення з підпрограми, якщо мінус Повернення з підпрограми, якщо є перенесення Повернення з підпрограми, якщо немає перенесення Повернення з підпрограми, якщо парний парітет Повернення з підпрограми, якщо непарний парітетRST 0 RST 1 RST 2 RST 3 RST 4 RST 5 RST 6 RST 7C7 CF D7 DF E7 EF F7 FFПовторний запуск програми з адреси 00H Повторний запуск програми з адреси 08H Повторний запуск програми з адреси 10H Повторний запуск програми з адреси 18H Повторний запуск програми з адреси 20H Повторний запуск програми з адреси 28H Повторний запуск програми з адреси 30H Повторний запуск програми з адреси 38HSPHL F9Завантажити SP з HLSHLD *aa*22Помістити HL в пам'ять за адресою *aa*STA *aa*32Помістити A в пам'ять LOC за адресою *aa*STAX B02Помістити A в пам'ять LOC (BC)STAX D12Помістити A в пам'ять LOC (DE)STC37Встановити індикатор перенесенняSUB A SUB B SUB C SUB D SUB E SUB H SUB L97 90 91 92 93 94 95Відняти A від A (очистити акумулятор) Відняти B від A Відняти C від A Відняти D від A Відняти E від A Відняти H від A Відняти L від ASUB

M96Відняти вміст пам’яті LOC (HL) від ASUI υD6Відняти безпосередньо наступні дані υ від ASBB A SBB B SBB C SBB D SBB E SBB H SBB L9F 98 99 9A 9B 9C 9DВідняти A від A, очистити акумулятор Відняти із запозиченням B від A Відняти із запозиченням C від A Відняти із запозиченням D від A Відняти із запозиченням E від A Відняти із запозиченням H від A Відняти із запозиченням L від ASBB M9EВідняти із запозиченням вміст пам’яті LOC (HL) від ASBI υDEВідняти із запозиченням безпосередні дані υ від AXCHGEBОбмін вмісту пар регістрів DE і HLXCHLE3Обмін вершини стеку з вмістом пари регістрів HLXRA A XRA B XRA C XRA D XRA E XRA H XRA LAF A8 A9 AA AB AC ADЛогічна операція A АБО З ВИКЛЮЧЕННЯМ A (очищення A) Логічна операція B АБО З ВИКЛЮЧЕННЯМ A Логічна операція C АБО З ВИКЛЮЧЕННЯМ A Логічна операція D АБО З ВИКЛЮЧЕННЯМ A Логічна операція E АБО З ВИКЛЮЧЕННЯМ A Логічна операція H АБО З ВИКЛЮЧЕННЯМ A Логічна операція L АБО З ВИКЛЮЧЕННЯМ AXRA MAEВміст пам’яті LOC (HL) АБО З ВИКЛЮЧЕННЯМ AXRI υ EEБезпосередні дані υ АБО З ВИКЛЮЧЕННЯМ A

Д1. 1. Символи і скорочення

Наступні символи і скорочення використовуються в описах команд мікропроцесора I8080.

Символ	**Значення**
A	Акумулятор (регістр A)
Addr	Значення адреси - 16 біт
Data	8-розрядне число (дані)
data 16	16-розрядне число (дані)
byte 2	Другий байт команди
byte 3	Третій байт команди
port	8-розрядна адреса пристрою уведення-виведення
r, r1, r2	Один із регістрів A, B, C, D, E, H, L
DDD, SSS	Набір бітів, який присвоєний одному із регістрів A, B, C, D, E, H, L (DDD - призначення, SSS - джерело)

DDD SSS	Регістр
111	A
000	B
001	C
010	D
011	E
100	H
101	L

rp	Одна з пар регістрів: B представляє пару BC, де B - старший регістр, C - молодший регістр; D представляє пару DE, де D - старший регістр, E - молодший регістр; H - представляє пару HL, де H - старший регістр, L - молодший регістр; SP - 16-розрядний показник стеку.
RP	Набір бітів, який присвоєний одній із пар регістрів B, D, H, SP RP Пара регістрів 00 BC 01 DE 10 HL 11 SP
rh	Перший регістр (який містить старший байт) даної пари
rl	Другий регістр (який містить молодший байт) даної пари
PC	16-розрядний регістр лічильника команд (PCH і PCL представляють відповідно 8 старших і 8 молодших біт)
SR	16-розрядний регістр покажчика стеку (SPH і SPL представляють відповідно 8 старших і 8 молодших біт)
rm	m-біт регістра (біти номеровані від 0 до 7 справа наліво)
LABEL	16-розрядна адреса підпрограми
	Індикатори
Z	Нуль
S	Знак
P	Парність
CY	Перенесення
AC	Додаткове перенесення
( )	Вміст комірки пам’яті або регістра, ім’я якого або адреса, які поміщені в дужках
←	“Переміщується в”
∧	Логічне множення (кон’юнкція), логічна операція І
⊕	Логічна операція АБО З ВИКЛЮЧЕННЯМ
∨	Логічне додавання (диз’юнкція), логічна операція АБО
+	Додавання
-	Віднімання
*	Множення
↔	“Замінити на”
-	Обернений код
n	Номер повторного запуску (від 0 до 7)
NNN	Двійкове представлення (000 -111) номера повторного запуску

Д1. 2. Команди передачі даних

Команди цієї групи служать для переміщення даних між регістрами, а також між регістрами і комірками пам’яті. Індикатори умови не встановлюються командами цієї групи.

MOV r1, r2 (Move register). Передача між регістрами. (r1)←(r2). Вміст регістра 2 передається в регістр 1.

0 1 D D D S S S
Циклів - 1; періодів T - 5, адресація - регістрова.

MOV r, M (Move from memory). Передача з пам’яті (r)←((H)(L)). Вміст пам’яті, адреса якої знаходиться в парі HL, передається в регістр r.

0 1 D D D 1 1 0
Циклів - 2; періодів T - 7; адресація - непряма регістрова.

MOV M, r (Move to memory). Передача в пам’ять ((H)(L))←(r)). Вміст регістра r передається в пам’ять, адреса якої знаходиться в парі HL.

0 1 1 1 0 S S S
Циклів - 2; періодів T - 7; адресація - непряма регістрова.

MVI r, data (Move immediate). Передача безпосередня. (r)←(2 байт). Вміст байта 2 команди передається в регістр r.

0 0 D D D 1 1 0Дані
Циклів - 2; періодів T - 7; адресація - безпосередня.

MVI M, data (Move to memory immediate). Передача в пам’ять безпосередня. ((H)(L))←(2 байт). Вміст байта 2 команди передається в пам’ять, адреса якої вказана парою HL.

0 0 1 1 0 1 1 0Дані
Циклів - 3; періодів T - 10; адресація безпосередня, непряма регістрова.

LXI rp, data 16 (Load register pair immediate). Завантажити безпосередньо пару регістрів. (rh)←(байт 3); (rl)←(2 байт). Байт 3 команди передається в старший регістр (rh), пари rp, а байт 2 - в молодший регістр (rl) тієї ж пари.

0 0 R P 0 0 0 1Молодший байт данихСтарший байт даних
Циклів - 3; періодів T - 10; адресація - безпосередня.

LDA addr (Load accumulator direct). Пряме завантаження акумулятора (A)←((байт 3)(байт 2)). Вміст пам’яті, адреса якої визначена байтами 2 і 3 команди, передається в акумулятор (регістр A).

0 0 1 1 1 0 1 0Молодший байт адресиСтарший байт адреси

Циклів - 4; періодів Т - 13; адресація - пряма.

STA addr (Store accumulator direct). Пряме розміщення вмісту акумулятора. ((байт 3)(байт 2))←(A). Вміст акумулятора передається в пам'ять, адреса якої вказана байтами 2 і 3 команди.

0 0 1 1 1 0 1 0Молодший байт адресиСтарший байт адреси
Циклів - 4; періодів Т - 13; адресація - пряма.

LHLD addr (Load H and L direct). Пряме завантаження H і L. (L)←((байт 3)(байт 2)); (H)←((байт 3)(байт 2)+1). Вміст комірки пам'яті, адреса якої визначається байтами 2 і 3 команди, передається в регістр L. Вміст наступного байту пам'яті передається в регістр H.

0 0 1 0 1 0 1 0Молодший байт адресиСтарший байт адреси

Циклів - 5; періодів Т - 16; адресація - пряма.

SHLD addr (Store H ahd L direct). Помістити H і L прямо. (байт 3)(байт 2))←(L); ((байт 3)(байт 2)+1)←(H). Вміст регістра L передається в комірку пам'яті, адреса якої визначається байтами 2 і 3 команди. Вміст регістру H передається в наступний байт пам'яті.

0 0 1 0 0 0 1 0Молодший байт пам'ятіСтарший байт пам'яті

Циклів - 5; періодів Т - 16; адресація - пряма.

LDAX rp (Load accumulator inderect). Завантажити акумулятор непрямо. (A)←((rp)). Вміст комірки пам'яті, адреса якої визначається парою регістрів rp, передається в регістр A. Можуть використовуватись тільки пари rp=B (регістри B і C) або rp=D (регістри D і E).

0 0 R P 1 0 1 0

Циклів - 2; періодів Т - 7; адресація - непряма регістрова.

STAX rp (Store accumulator inderect). Завантажити регістр непрямо. (rp))←(A). Вміст акумулятора A передається в комірку пам'яті, адреса якої міститься в парі регістрів rp. Можуть використовуватись тільки пари rp=B (регістри B і C) або rp=D (регістри D і E).

0 0 R P 0 0 1 0
Циклів - 2; періодів Т - 7; адресація - непряма регістрова.

XCHG (Exchange H and with D and E). Обмін H і L з D і E (H)↔(D); (L)↔(E). Вміст регістрів H і L обмінюється з вмістом регістрів D і E.

1 1 1 0 1 0 1 1
Циклів - 1; періодів Т - 4; адресація - регістрова.

Д1. 3. Арифметичні команди

Арифметичні команди призначені для виконання операцій додавання, додавання з перенесенням, віднімання, віднімання із запозиченням, інкрементування, декрементування, десяткової корекції акумулятора.

Ці команди оперують з даними в пам'яті і регістрах. У всіх випадках, окрім вказаних виключень, встановлюються індикатори нуля Z, знаку S, парності P, перенесення CY і додаткового перенесення AC. Всі операції віднімання проводяться з використанням доповнювального коду.

ADD r (ADD register). Додавання вмісту регістра. (A)←(A)+(r). Вміст регістра r додається до вмісту акумулятора. Результат розміщується в акумуляторі.

1 0 0 0 0 S S S

Циклів - 1; періодів T - 4; адресація - регістрова; індикатори - Z, S, P, CY, AC.

ADD M (Add memory). Додавання даних, які знаходяться у пам'яті. (A)←(A)+((H)(L)). Вміст пам'яті, адреса якої знаходиться в регістрах H і L, додається з вмістом акумулятора. Результат розміщується в акумуляторі.

1 0 0 0 0 1 1 0

Циклів - 2; періодів T - 7; адресація - непряма регістрова; індикатори - Z, S, P, CY, AC.

ADI data (Add immediate). Безпосереднє додавання. (A)←(A)+(байт 2). Вміст байта команди 2 додається до вмісту акумулятора. Результат розміщується в акумуляторі.

1 1 0 0 0 1 1 0

Циклів - 2; періодів T - 7; адресація - безпосередня; індикатори - Z, S, P, CY, AC.

ADC r (Add register with carry). Додавання вмісту регістра і перенесення. (A)←(A)+(r)+(CY). Вміст регістра r та індикатора перенесення (біт переповнення) додаються до вмісту акумулятора. Результат розміщується в акумуляторі.

1 0 0 0 1 S S S

Циклів - 1; періодів T - 4; адресація - регістрова; індикатори - Z, S, P, CY, AC.

ADC M (Add memory with carry). Додавання вмісту пам'яті і перенесення. (A)←(A)+((H)(L))+(CY). Вміст пам'яті, адресою якої є вміст пари регістрів HL, і індикатора перенесення додається до вмісту акумулятора. Результат розміщується в акумуляторі.

1 0 0 0 1 1 1 0

Циклів - 2; періодів T - 7; адресація - непряма регістрова; індикатори - Z, S, P, CY, AC.

ACI data (Add immediate with carry). Безпосереднє додавання з врахуванням перенесення. (A)←(A)+(байт 2)+(CY). Вміст 2 байт команди і індикатора перенесення додається до вмісту акумулятора. Результат розміщується в акумуляторі.

1 1 0 0 1 1 1 0

Циклів - 2; періодів T - 7; адресація - безпосередня; індикатори - Z, S, P, CY, AC.

SUB r (Subtract register). Віднімання вмісту регістра. (A)←(A)-(r). Вміст регістра r віднімається від вмісту акумулятора. Результат розміщується в акумуляторі.

1 0 0 1 0 S S S

Циклів - 1; періодів T - 5; адресація - регістрова; індикатори - Z, S, P, CY, AC.

SUB M (Subtract memory). Віднімання вмісту пам'яті. (A)←(A)-((H)(L)). Вміст пам'яті, адреса якої є вмістом пари регістрів HL, віднімається від вмісту акумулятора. Результат розміщується в акумуляторі.

1 0 0 1 0 1 1 0

Циклів - 2; періодів T - 7; адресація - непряма регістрова; індикатори - Z, S, P, CY, AC.

SUI data (Subtract immediate). Безпосереднє віднімання. (A)←(A)-(байт 2). Вміст 2-го байту команди віднімається від вмісту акумулятора. Результат розміщується в акумуляторі

1 1 0 1 0 1 1 0Дані

Циклів - 2; періодів T - 7; адресація - безпосередня; індикатори - Z, S, P, CY, AC.

SBB r (Subtract register with borrow). Віднімання вмісту регістра і перенесення. (A)←(A)-(r)-(CY). Вміст регістра r і індикатора CY віднімаються від вмісту акумулятора. Результат розміщується в акумуляторі.

1 0 0 1 1 S S S

Циклів - 1; періодів T - 4; адресація - регістрова; індикатори - Z, S, P, CY, AC.

SBB M (Subtract memory with borrow). Віднімання вмісту пам'яті і перенесення. (A)←(A)-((H)(L))-(CY). Вміст пам'яті, адресою якої є вміст пари регістрів HL, і індикатора CY віднімаються від вмісту акумулятора. Результат розміщується в акумуляторі.

1 0 0 1 1 1 1 0

Циклів - 2; періодів T - 7; адресація - непряма регістрова; індикатори - Z, S, P, CY, AC.

SBB data (Subtract immediate with borrow). Безпосереднє віднімання даних і перенесення. (A)←(A)-(байт 2)-(CY). Вміст 2-го байту команди і індикатора CY віднімаються від вмісту акумулятора. Результат розміщується в акумуляторі.

1 1 0 1 1 1 1 0Дані

Циклів - 2; періодів T - 7; адресація - безпосередня; індикатори - Z, S, P, CY, AC.

INR r (Increment register). Інкремент вмісту регістра. (r)←(r)+1. Вміст регістра збільшується на 1. Встановлюються всі індикатори ознак, за вийнятком CY.

0 0 D D D 1 0 0

Циклів - 1; періодів T - 4; адресація - регістрова; індикатори - Z, S, P, AC.

INR M (Increment memory). Інкремент вмісту пам'яті. ((H)(L))←((H)(L))+1. Вміст пам'яті, адреса якої розміщена в парі регістрів HL, збільшується на 1. Встановлюються всі індикатори ознак, за вийнятком CY.

0 0 1 1 0 1 0 0
Циклів - 3; періодів T - 10; адресація - непряма регістрова; індикатори - Z, S, P, AC.

DCR r (Decrement register). Декремент вмісту регістра. (r)←(r)-1. Вміст регістра r зменшується на 1. Встановлюються всі індикатори, за вийнятком CY.

0 0 D D D 1 0 1
Циклів - 1; періодів T - 4; індикатори - Z, S, P, AC.

DCR M (Decrement memory). Декремент вмісту пам'яті. ((H)(L))←((H)(L))-1. Вміст пам'яті, адреса якої знаходиться в парі регістрів HL, зменшується на 1. Встановлюються всі індикатори, за вийнятком CY.

0 0 1 1 0 1 0 1
Циклів - 3; періодів T - 10; адресація - непряма регістрова; індикатори - Z, S, P, AC.

INX rp (Increment register pair). Інкремент вмісту пари регістрів (rh)(rl)←(rh)(rl)+1. Вміст пари регістрів rp збільшується на 1. Не встановлюються ніякі індикатори.

0 0 R P 0 0 1 1
Циклів - 1; періодів T - 6; адресація - регістрова; індикатори не змінюються.

DCX rp (Decrement register pair). Декремент вмісту пари регістрів (rh)(rl)←(rh)(rl)-1. Вміст пари регістрів rp зменшується на 1. Не встановлюються ніякі індикатори.

0 0 R P 1 0 1 1
Циклів - 1; періодів T - 6; адресація - регістрова; індикатори не змінюються.

DAD rp (Add register pair to H and L). Додати вміст пари регістрів до вмісту пари регістрів HL.(H)(L)←(H)(L)+(rh)(rl). Вміст пари регістрів rp додається до вмісту пари HL. Встановлюються тільки індикатор CY. Він встановлюється в 1, якщо є перенесення при додаванні з подвоєною точністю, якщо ні - встановлюється в 0.

0 0 R P 1 1 0 1
Циклів - 3; періодів T - 10; адресація - регістрова; індикатор - CY.

DAA (Decimal adjust accumulator). Десяткова корекція акумулятора. 8-и розрядне число в акумуляторі розбивається на два 4-х розрядні двійково-десяткові. Далі виконуються наступні дії: 1) якщо значення молодшої тетради акумулятора більше 9 або встановлений індикатор AC, то до вмісту акумулятора додається 6; 2) якщо значення старшої тетради більше 9 або встановлений індикатор перенесення CY, то 6 додається до значення старшої тетради акумулятора.

0 0 1 0 0 1 1 1
Циклів - 1; періодів T - 4; індикатори - Z, S, P, CY, AC.

Д1. 4. Логічні команди

Завданням цих команд являється виконання логічних операцій І, АБО, АБО З ВИКЛЮЧЕННЯМ, порівняння, переміщення та інвертування. Ці команди виконують логічні операції над даними у пам'яті або регістрах та індикаторах. Позначення у скороченнях відповідають параграфу Д1.1.

ANA r (AND register). Логічна операція І акумулятора з регістром. (A)←A)∧(r). Логічне множення вмісту регістра r і вмісту акумулятора по-бітно. Результат розміщується в акумуляторі. Ознака перенесення CY встановлюється в 0, а ознака допоміжнього перенесення AC приймає значення результату логічної операції АБО, виконаної із третіми бітами операндів.

1 0 1 0 0 S S S

Циклів - 1; періодів T - 4; адресація - регістрова; індикатори - Z, S, P, CY, AC.

ANA M (AND memory). Логічна операція І акумулятора і пам'яті. (A)←(A)∧((H)(L)). Логічне множення вмісту комірки пам'яті, адреса якої знаходиться в регістрах HL, із вмістом акумулятора. Результат розміщується в акумуляторі. Ознака перенесення CY встановлюється в 0, а ознака допоміжнього перенесення AC приймає значення результату логічної операції АБО, виконаної із третіми бітами.

1 0 1 0 0 1 1 0

Циклів - 2; періодів T - 7; адресація - непряма регістрова; індикатори - Z, S, P, CY, AC.

ANI data (AND immediate). Логічна операція І акумулятора та безпосередньо наступних даних. (A)←(A)∧(байт 2). Логічне множення вмісту 2-го байта команди із вмістом акумулятора. Результат розміщується в акумуляторі. Ознака перенесення CY встановлюється в 0, а ознака допоміжнього перенесення AC приймає значення результату логічної операції АБО, виконаної із третіми бітами.

1 1 1 0 0 1 1 0Дані

Циклів - 2; періодів T - 7; адресація - безпосередня; індикатори - Z, S, P, CY, AC.

XRA r (Exclusive OR register). Логічна операція сума по модулю два (АБО З ВИКЛЮЧЕННЯМ) акумулятора та регістра. (A)←(A)⊕(r). АБО З ВИКЛЮЧЕННЯМ виконується із вмістом регістра r і акумудятора. Результат розміщується в акумуляторі. Ознака перенесення CY та ознака допоміжнього перенесення AC встановлюються в 0.

1 0 1 0 1 S S S

Циклів - 1; періодів T - 4; адресація - регістрова; індикатори - Z, S, P, CY, AC.

XRA M (Exclusive OR memory). Логічна операція сума по модулю два (АБО З ВИКЛЮЧЕННЯМ) акумулятора та пам'яті. (A)←(A)⊕((H)(L)). АБО З ВИКЛЮЧЕННЯМ виконується із вмістом комірки пам'яті, адреса якої вказана парою HL, і акумудятора.

Результат розміщується в акумуляторі. Ознака перенесення CY та ознака допоміжного перенесення AC встановлюються в 0.

1 0 1 0 1 1 1 0
Циклів - 2; періодів T - 7; адресація - непряма регістрова; індикатори - Z, S, P, CY, AC.

XRI data (Exclusive OR immediate). Логічна операція сума по модулю два (АБО З ВИКЛЮЧЕННЯМ) акумулятора та безпосередньо наступних даних. (A)←(A)⊕(байт 2). АБО З ВИКЛЮЧЕННЯМ виконується із вмістом 2-го байта команди і акумулятора. Результат розміщується в акумуляторі. Ознака перенесення CY та ознака допоміжного перенесення AC встановлюються в 0.

1 1 1 0 1 1 1 0Дані
Циклів - 2; періодів T - 7; адресація - безпосередня; індикатори - Z, S, P, CY, AC.

ORA r (OR register). Логічна операція АБО акумулятора з регістром. (A)←A)∨(r). Логічне додавання вмісту регістра r і вмісту акумулятора по-бітно. Результат розміщується в акумуляторі. Ознака перенесення CY та ознака допоміжного перенесення AC встановлюються в 0.

1 0 1 1 0 S S S
Циклів - 1; періодів T - 4; адресація - регістрова; індикатори - Z, S, P, CY, AC.

ORA M (OR memory). Логічна операція АБО акумулятора з пам’ятю. (A)←A)∨ ((H))(L)). Логічне додавання вмісту комірки пам’яті, адреса якої вказана парою HL, і вмісту акумулятора по-бітно. Результат розміщується в акумуляторі. Ознака перенесення CY та ознака допоміжного перенесення AC встановлюються в 0.

1 0 1 1 0 1 1 0
Циклів - 2; періодів T - 7; адресація непряма регістрова; індикатори - Z, S, P, CY, AC.

ORI data (OR immediate). Логічна операція АБО акумулятора безпосередньо наступних даних. (A)←A)∨(байт 2). Логічне додавання із вмістом 2-го байту команди і вмісту акумулятора. Результат розміщується в акумуляторі. Ознака перенесення CY та ознака допоміжного перенесення AC встановлюються в 0.

1 1 1 1 0 1 1 0Дані
Циклів - 2; періодів T - 7; адресація - безпосередня; індикатори - Z, S, P, CY, AC.

CMP r (Compare register). Порівняти аккумулятор з регістром. (A)-(r). Вміст регістру r віднімається від вмісту акумулятора. Вміст акумулятора не змінюється. Індикатори встановлюються, як при відніманні. Ознака нуля Z встановлюється в 1, якщо (A)=(r), а ознака перенесення CY встановлюється в 1, якщо (A)<(r).

1 0 1 1 1 S S S
Циклів - 1; періодів T - 4; адресація - регістрова; індикатори - Z, S, P, CY, AC.

CMP M (Compare memory). Порівняти аккумулятор з пам’ятю. (A)-((H)(L)). Вміст пам’яті, адреса якої являється вмістом пари HL, віднімається від вмісту акумулятора. Вміст акумулятора не змінюється. Ознака нуля Z встановлюється в 1, якщо (A)=((H)(L)), а ознака перенесення CY встановлюється в 1, якщо (A)<((H)(L)).

1 0 1 1 1 1 1 0
Циклів - 2; періодів T - 7; адресація - непряма регістрова; індикатори - Z, S, P, CY, AC.

CPI data (Compare immediate). Порівняти аккумулятор з безпосередньо наступними даними. (A)-(байт 2). Вміст 2-го байту команди віднімається від акумулятора. Вміст акумулятора не змінюється. Індикатори встановлюються, як при відніманні. Ознака нуля Z встановлюється в 1, якщо (A)=(байт 2), а ознака перенесення CY встановлюється в 1, якщо (A)<(байт 2).

1 1 1 1 1 1 1 0Дані
Циклів - 2; періодів T - 7; адресація - безпосередня; індикатори - Z, S, P, CY, AC.

RLC (Rotate left). Переміщення ліворуч. (bn+1)←(bn); (b0)←(b7); (CY)←(b7). Вміст акумулятора переміщується на одну позицію ліворуч. Молодший біт b0 і біт перенесення CY приймають значення витісненого біта, тобто того, що був старшим бітом b7. Встановлюється лише біт перенесення CY.

0 0 0 0 0 1 1 1
Циклів - 1; періодів T - 4; індикатор - CY.

RRC (Rotate right). Переміщення праворуч. (bn)←(bn+1); (b7)←(b0); (CY)←(b0). Вміст акумулятора переміщується на одну позицію праворуч. Старший біт b7 і біт перенесення CY приймають значення витісненого біта, тобто того, що був молодшим бітом b0. Встановлюється лише біт перенесення CY.

0 0 0 0 0 1 1 1
Циклів - 1; періодів Т - 4; індикатор - CY.

RAL (Rotate left trough carry). Циклічне переміщення ліворуч. (bn+1)←(bn); (CY)←(b7); (b0)←(CY). Вміст акумулятора переміщується на одну позицію ліворуч разом з бітом перенесення CY. В молодшому біті b0 встановлюється вміст CY, а біт перенесення CY приймає значення витісненого біта, тобто того, що був старшим бітом b7. Встановлюється лише біт перенесення CY.

0 0 0 1 0 1 1 1
Циклів - 1; періодів Т - 4; індикатор - CY.

RAR (Rotate right trough carry). Циклічне переміщення праворуч. (bn)←(bn+1); (CY)←(b0); (b7)←(CY). Вміст акумулятора переміщується на одну позицію праворуч разом із бітом перенесення CY. Старший біт b7 приймає значення біта перенесення CY, а CY - значення витісненого біта, тобто того, що був молодшим бітом b0. Встановлюється лише біт перенесення CY.

0 0 0 1 1 1 1 1
Циклів-1; періодів Т - 4; індикатор - CY.

CMA (Complement accumulator). Інвертувати акумулятор. $(\mathrm{A}) \leftarrow (\overline{\mathrm{A}})$. Вміст акумулятора інвертується (якщо біт мав значення 0, то він приймає значення 1, якщо біт мав значення 1, то він приймає значення 0). Індикатори не встановлюються.

0 0 1 0 1 1 1 1
Циклів - 1; періодів Т - 4; індикатори - ніякі.

CMC (Complement carry). Інвертувати біт перенесення. $(\mathrm{CY}) \leftarrow (\overline{\mathrm{CY}})$. Інвертується вміст індикатора CY. Ніякі інші індикатори не встановлюються.

0 0 1 1 1 1 1 1

Циклів - 1; періодів Т - 4; індикатори - ніякі.

STC (Set carry). Встановити біт перенесення. (CY)←1. Ознака перенесення CY приймає значення 1. Ніякі інші індикатори не встановлюються.

0 0 1 1 0 1 1 1
Циклів - 1; періодів Т - 4; індикатори - ніякі.

Д1. 5. Команди переходів

Ця група команд вміщує команди переходу, виклику, повернення і повторного запуску (рестарту). Всі команди даної групи змінюють послідовний (нормальний) хід програми. Тут є команди двох типів: умовного і безумовного переходів. Безумовні переходи просто виконують операцію, визначену лічильником команд; умовні - перевіряють стан одного із індикаторів регістру ознак мікропроцесора для визначення доцільності розгалуження в програмі. Умови, які перевіряються, записуються у слідуючій формі:

УмоваІндикаторCCCNZ - не нульZ=0000Z - нульZ=1001NC - немає перенесенняCY=0010C - перенесенняCY=1011PO - непарністьP=0100PE - парністьP=1101P - плюсS=0110M - мінусS=1111

Зміст скорочень, які використовуються у описах команд, приведений у параграфі Д.1.

JMP addr (Jump). Перехід (розгалуження). (PC)←(байт 3)(байт 2). Керування передається команді, адреса якої вказана у байтах 2 і 3 даної команди.

1 1 0 0 0 0 1 1Молодший байт адресиСтарший байт адреси
Циклів - 3; періодів Т - 10; адресація - безпосередня; індикатори - ніякі.

Jcondition addr (Conditionel jamp). Умовне розгалуження. Якщо (CCC), то (PC)←(байт 3)(байт 2). Якщо встановлена умова виконується, то керування передається команді, адреса якої вказана у байтах 2 і 3 даної команди; якщо ні - виконується наступна, за даною, команда.

1 1 C C C 0 1 0Молодшй байт адресиСтарший байт адреси
Циклів - 3; періодів Т - 10; адресація - безпосередня; індикатори - ніякі.

CALL addr (Call). Виклик. ((SP)-1)←(PCH); ((SP)-2)←(PCL); (SP)←(SP)-2; (PC)←(байт 3)(байт 2). Старших 8 біт лічильника команд, які вказують адресу слідуючої команди передаються в пам’ять, адреса якої на 1 менше вмісту регістру SP. Молодших вісім біт лічильника команд, які вказують адресу слідуючої команди передаються в пам’ять, адреса якої на 2 менше вмісту регістру SP. Вміст регістру SP двічі

декрементується. Керування передається команді, адреса якої визначається байтами 3 і 2 поточної команди.

1 1 0 0 1 1 0 1Молодший байт адресиСтарший байт адреси
Циклів - 5; періодів Т - 17; адресація - безпосередня, непряма регістрова; індикатори - ніякі.

Condition addr (Conditionel call). Умовний виклик. Якщо (CCC), то ((SP)-1)←(PCH); ((SP)-2)←(PCL); (SP)←(SP)-2; (PC)←(байт 3)(байт 2).

Якщо задані умови виконуються, виконуються дії, визначені в команді CALL (див. вище); якщо ні - виконується слідуюча, після даної, команда.

1 1 C C C 1 0 0Молодший байт адресиСтарший байт адреси
Циклів - 3/5; періодів Т - 11/17; адресація - безпосередня, непряма регістрова; індикатори - ніякі.

RET (Return). Повернення. (PCL)←((SP)); (PCH)←(SP))+1); (SP)←(SP)+2. Вміст пам’яті, адреса якої визначена в регістрі SP, передається в молодших 8 біт регістру PC. Вміст пам’яті, адреса якої на 1 старша вмісту регустру SP, передається в старших 8 біт регістру PC. Вміст регістру SP двічі інкрементується.

1 1 0 0 1 0 0 1
Циклів - 3; періодів Т - 10; адресація - непряма регістрова; індикатори - ніякі.

Rcondition (Conditional return). Повернення умовне. Якщо (CCC), то (PCL)←(SP)); (PCH)←((SP)+1); (SP)←(SP)+2. Якщо дана умова виконана, виконуються дії, визначені командою повернення (див. вище), якщо ні - виконується слідуюча, після даної, команда.

1 1 C C C 0 0 0
Циклів - 1/3; періодів Т - 5/11; адресація - непряма регістрова; індикатори - ніякі.

RST (Restart). Повторний пуск (рестарт). ((SP)-1←(PCH); ((SP)-2←(PCL); (SP)←(SP)-2; (PC)←8X(NNN). Старші 8 біт адреси наступної команди передаються в пам’ять, адреса якої на 1 менша вмісту SP. Молодші 8 біт адреси наступної команди передаються в пам’ять, адреса якої менша на 2 одиниці вмісту регістру SP. Вміст регістру SP декрементується двічі. Керування передається команді, адресою якої є 8 разів вміст NNN (8xNNN).

1 1 N N N 1 1 1
Циклів - 3; періодів Т - 11; адресація - непряма регістрова; індикатори - ніякі.

Лічильник команд після команди рестарту.

PCHL (Jump H and L indirect - move H and L to PC). Перейти H і L непрямо - передати H і L в PC. (PCH)←(H); (PCL)←(L). Вміст регістру H передається в старших 8 біт регістру PC, а регістру L - в молодших 8 біт регістру PC.

1 1 1 0 1 0 0 1

Циклів - 1; періодів T - 5; адресація - регістрова; індикатори - ніякі.

Д1. 6. Команди стеку, уведення-виведення та керування

Розглянемо короткий опис цих команд. Вони виконують операції запису у стек і читання з нього, уведення і виведення даних, обміну даними, підтвердження і непідтвердження переривань. Команди цієї групи не змінюють вміст регістра ознак, крім тих випадків, які будуть окремо описані.

PUSH rp (Push). Записати у стек вміст пари регістрів. ((SP)-1)←(rh); ((SP)-2)←(rl); ((SP)←(SP)-2). Вміст старшого регістру пари rp передається у пам'ять, адреса якої менше на 1 вмісту регістру SP. Вміст молодшого регістру пари rp передається в пам'ять, адреса якої менше на 2 вмісту регістру SP. Вміст регістру SP декрементується двічі. Пара регістрів rp=SP не може бути вказана у форматі цієї команди.

1 1 R P 0 1 0 1

Циклів - 3; періодів T-12; адресація - непряма регістрова; індикатори - не змінюються.

PUSH PSW (Push processor status word). Розмістити в стеку слово стану процесора. ((SP)-1)←(A); ((SP)-2)0←(CY); ((SP)-2)1←X; ((SP)-2)2←(P); ((SP)-2)3←X; ((SP)-2)4←(AC); ((SP)-2)5←X; ((SP)-2)6←(Z); ((SP)-2)7←(S); (SP)←(SP)-2. X - значення біту не визначене. Вміст регістру A передається в пам'ять, адреса якої менше на 1 вмісту регістру SP. Вміст регістру ознак збирається в слові стану процесора і передається по-бітно (див. індекси в мнемоніці операцій) у пам'ять, адреса якої менше на 2 вмісту регістру SP, який декрементується двічі.

1 1 1 1 0 1 0 1

Циклів - 3; періодів T - 11; адресація - непряма регістрова; індикатори - не змінюються.

Слово стану процесора (PSW):

POP rp (Pop). Завантаження із стеку регістрів. (rl)←((SP)); (rh)←((SP)+1); (SP)←(SP)+2. Вміст пам'яті, адреса якої визначається регістром SP, передається в молодший регістр пари rp. Вміст пам'яті, адреса якої більше на 1 вмісту регістру SP, передається у старший регістр пари rp. Вміст регістру SP інкрементується двічі. Пара регістрів rp=SP не може бути вказана у форматі цієї команди.

1 1 R P 0 0 0 1
Циклів - 3; періодів T - 10; адресація - непряма регістрова; індикатори - не змінюються.

POP PSW (Pop processor status word). Прочитати із стеку слово стану процесора. (CY)←((SP))0; (P)←((SP))2; (AC)←((SP))4; (Z)←((SP))6; (S)←((SP))7; (A)←((SP))+1; (SP)←(SP)+2. Вміст пам'яті, адреса якої визначена вмістом регістру SP, по-бітно (відповідно з індексами в записі операцій) використовується для відновлення індикаторів умови. Вміст пам'яті, адреса якої більше на 1 вмісту SP, передається в регістр A. Вміст SP інкрементується двічі.

1 1 1 1 0 0 0 1
Циклів - 3; періодів T - 10; адресація - непряма регістрова; індикатори - Z, S, P, AC, CY.

XTHL (Exchange stack top with H and L). Обміняти вершину стеку із H і L. (L)←((SP)); (H)←((SP)+1). Вміст регістру L обмінюється із вмістом пам'яті, адреса якої визначена вмістом регістру SP. Вміст регістру H обмінюється із вмістом пам'яті, адреса якої більше на 1 вмісту SP.

1 1 1 0 0 0 1 1
Циклів - 5; періодів T - 18; адресація - непряма регістрова; індикатори - ніякі.

SPHL (Move HL to SP). Передати HL у SP. (SP)←(H)(L). Вміст регістрів H і L (16 біт) передається у регістр SP.

1 1 1 1 1 0 0 1
Циклів -1; періодів T - 5; адресація - регістрова; індикатори - ніякі.

IN port (Input). Увести дані. (A)←(дані). Дані, які розміщені на 8-розрядній двонапрямні шині даних певним портом, передаються у регістр A.

1 1 0 1 1 0 1 1Адреса порта
Циклів - 3; періодів T - 10; адресація - пряма; індикатори - ніякі.

OUT port (Output). Вивести дані. (Дані)←(A). Вміст регістру A виводиться на 8-розрядну двонапрямну шину даних для передачі у певний порт.

1 1 0 1 0 0 1 1Адреса порта
Циклів - 3; періодів T - 10; адресація - пряма; індикатори - ніякі.

EI (Enable interrupts). Дозвіл переривань. Переривання дозволяються відразу після виконання слідуючої команди. Під час виконання команди EI переривання заборонені.

1 1 1 1 1 0 1 1
Циклів - 1; періодів T - 4; індикатори - ніякі.

DI (Disable interrupts). Заборона переривань. Перериваня забороняються відразу після виконання команди DI. Під час виконання команди DI переривання заборонені.

1 1 1 1 0 0 1 1
Циклів - 1; періодів T - 4; індикатори - ніякі.

HLT (Halt). Зупинка. Процесор зупиняється. Вміст регістрів та індикатори не змінюються.

0 1 1 1 0 1 1 0
Циклів - 1; періодів T - 5 індикатори - не змінюються.

NOP (No op). Операція відсутня. Не виконується ніяка операція. Регістри та індикатори не змінюються.

0 0 0 0 0 0 0 0
Циклів - 1; періодів T - 4; індикатори - не змінюються.

Додаток 2.

Інструкція до Редактора текстів та Асемблера ПЕОМ ПК-01 "Львів"

Редактор текстів розміщується в ОЗПр з адреси 9F00H, кінцева адреса AFA0H. Деректива Монітора, яка запускає програму - G9F00.

Курсор переміщується по екрану за допомогою клавіш: стрілка униз, уверх, управо, уліво, на початок екрану.

Призначення клавіш:

<R> - поміняти місцями рядки;

<B> - зробити копію рядка;

< > - зсув частини рядка управо;

<CD> - зсув частини рядка уліво;

<ПЧ> - розсунути рядки для вставки;

<П/Д> - зсув рядків;

<ЧМЛ> - перемістити сторінку уверх;

<BS> - перемістити сторінку униз;

<ЗМЛ> - на початок тексту;

<ED> - на кінець тексту, занести сторінку у пам`ять;

Клавіша <F0> використовується в комбінації з такими клавішами:

<F0><+> - зітерти текст;

<F0><5> - звільнити місце для вставки (курсор встановити на початок рядка, посля якого потрібно зробити вставку);

<F0><2> - записати текст файл;

<AS> - виконати вказівки, які задані командами <F0>.

Програма Асемблер розміщується з адреси 0020H по 2421H. Завантажується програма клавішею <AS>, або дерективою Монітора - G2401.

Після завантаження на дисплеї повинно появитися повідомлення:

МАКРОАССЕМБЛЕР *ПК-01* ВЕР. 1.0.

P=

Після цього вказати:

1 - трансляція;

2 - лістінг;

3 - трансляція + об`єктний модуль;

4 - трансляція + об`єктний модуль + лістінг.

Після виконання вказаних деректив потрібно натиснути кнопку <СБР>, а після появи крапки у лівому верньому куті екрану вказати:

<R><0><ВК>.

При написані програм на мові Асемблера обов`язково після кожного набраного рядка натискувати кнопку <ВК>.

Додаток 3

Лістинг програми "EXPEREMENT"

```
10 GOSUB 860
30 GOSUB 1200
35 GOSUB 1000
40 DIM V(100),I(100)
50 POKE 73,0
60 GOSUB 300
65 E=10
70 CLS
72 PRINT"Уведіть знак вихідної напруги:"
74 PRINT
80 PRINT"знак ' + '  - на виході ЦАП додатній потенціал;"
84 PRINT
86 PRINT"знак ' - '   - на виході ЦАП від'ємний потенціал."
90 PRINT
92 INPUT A$
94 IF A$="+" THEN 100
96 IF A$="-" THEN 110
98 PRINT"Будьте уважні!"
99 GOTO 92
100 POKE 46302,4
104 GOTO 120
110 POKE 46302,12
115 GOTO 120
120 POKE 73,34
126 GOSUB 300
130 INPUT "Початкова напруга, В:";U1
134 INPUT "Кінцева напруга, В:";U2
136 PRINT"З яким кроком змінювати вихідну напругу?"
140 PRINT"Увага! Розмір кроку повинен бути більший за 0,1 В"
144 INPUT "Крок, В:";K
150 N1=INT(25.5*U1)
152 N2=INT(25.5*U2)
156 N3=INT(25.5*K)
158 CLS
159 J=0
160 FOR X=N1 TO N2 STEP N3
162 J=J+1
164 POKE 46300,X
166 POKE 46301,1
168 POKE 73,19
170 GOSUB 300
174 POKE 73,58
176 GOSUB 300
180 POKE 73,79
182 GOSUB 300
184 B=2
```

```
186 GOSUB 340
190 H=P*10+O+D*0.1+S*0.01
194 B=0
196 GOSUB 340
198 Q=P*10+O+D*0.1+S*0.01
200 V(J)=Q-H
202 I(J)=(H/R)*1000
204 PRINTJ;" U=";V(J);"B";"   I=";I(J);"MA"
210 NEXT X
214 POKE 73,0
216 GOSUB 510
220 PRINT"Графік будувати? (Y/N)"
222 INPUT A$
226 IF A$="Y" THEN 240
228 IF A$="N" THEN 236
230 CLS
231 GOTO 220
232 PRINT"Будьте уважні!"
236 STOP
240 CLS
245 PRINT"Куди виводити графік?"
247 PRINT"На дисплей - 1"
248 PRINT"На плотер  - 2"
250 INPUT A
252 IF A=1 THEN 700
254 IF A=2 THEN 800
256 CLS
258 PRINT"Будьте уважні!"
260 GOTO 245
300 POKE 74,180
310 X=USR(X)
320 RETURN
339 REM ------------ перетворення коду ------------
340 Z=PEEK(46304+B)
344 D=INT(Z/16)
346 S=Z-D*16
348 Y=PEEK(46305+B)
350 P=INT(Y/16)
352 O=Y-P*16
354 RETURN
499 REM ------------ управління пером ------------
500 POKE 46302,P
505 POKE 73,34
510 POKE 74,180
512 X=USR(X)
514 RETURN
519 REM ------------ вести лінію ------------
520 FOR F=A TO B STEP L
522 POKE 46300,F
524 POKE 46301,N
526 POKE 73,19
```

```
530 GOSUB 510
532 NEXT F
534 RETURN
539 REM ------------ вести лінію ------------
540 P=192
545 GOSUB 500
550 GOSUB 520
555 P=64
560 GOSUB 500
565 RETURN
569 REM ------------ іти в точку (X,Y) ------------
570 FOR W=1 TO 2
575 IF W=1 THEN POKE46300,Y
580 IF W=2 THEN POKE46300,X
585 POKE 46301,W
590 POKE 73,19
592 GOSUB 510
594 NEXT W
596 RETURN
599 REM ------------ затримка ------------
600 S=0
604 FOR D=0 TO E
606 S=S+Dï2
608 NEXT D
610 RETURN
619 REM ------------ рисувати вісі координат ------------
620 FOR T=1 TO 2
625 POKE 73,0
630 GOSUB 510
635 N=T:A=0:B=255:L=25.5
640 GOSUB 540
645 FOR I=51 TO 255 STEP 51
650 POKE 46300,I
652 POKE 46301,T
654 POKE 73,19
656 GOSUB 510
660 N=3-T:A=0:B=8:L=1
665 GOSUB 540
670 NEXT I
672 NEXT T
674 RETURN
700 CLS
705 FOR I=4 TO 18
710 PRINT
715 IF I=6 THEN PRINT TAB(6);"I,MA"
717 IF I=8 THEN PRINT TAB(1);"100"
720 IF I=11THEN PRINT TAB(2);"60"
722 IF I=15 THEN PRINT TAB(2);"20"
724 IF I=18 THEN PRINT TAB(4);"0     0,4    0,8  U,в"
726 NEXT I
728 PLOT 30,55,1
```

```
730 DRAW 150,55
732 FOR I=50 TO 130 STEP 20
734 PLOT I,55,1:DRAW I,59
736 NEXT I
738 PLOT 30,55,1
740 DRAW 30,190
742 FOR I=55 TO 155 STEP 20
744 PLOT 30,I,1:DRAW 34,I
746 NEXT I
750 FOR I=1 TO J
754 Y=INT(I(I))
756 X=INT(V(I)*100)
760 IF Y>=120 THEN Y=0
762 PLOT X+30,Y+55,1
764 NEXT I
766 STOP
800 GOSUB 1400
801 GOSUB 620
804 FOR I=1 TO J
806 X=INT(V(I)*255)
808 Y=INT(I(I)*2.55)
810 IF Y>=255 THEN Y=255
814 GOSUB 570
818 P=192
820 GOSUB 500
824 P=64
826 GOSUB 500
835 NEXT I
840 X=0:Y=0
845 GOSUB 570
850 STOP
860 A=PEEK(46098)
862 B=PEEK(46237)
863 IF A=B THEN RETURN
864 CLS
868 PRINT"Завантажте із магнітної стрічки драйвери "
870 PRINT" ЦАП і вольтметра."
872 POKE 73,0
874 POKE 74,248
876 X=USR(X)
1000 CLS
1010 PRINT"Струм через досліджувальний діод "
1020 PRINT" визначається шляхом вимірювання напруги "
1030 PRINT" на резисторі, увімкненому послідовно з діодом."
1049 GOTO 1102
1050 PLOT 50,160,1
1055 DRAW 110,160:DRAW 110,142
1057 DRAW 124,142:DRAW 124,138
1060 PLOT 90,160,1:DRAW 90,122
1062 DRAW 93,122:DRAW 93,108
1064 DRAW 86,108:DRAW 86,123
```

```
1066 DRAW 90,122
1068 PLOT 90,108,1:DRAW 90,100
1070 PLOT 50,100,1:DRAW 130,100
1072 DRAW 130,126:DRAW 140,126
1074 PLOT 120,140,1:DRAW130,134
1076 DRAW 140,134
1078 PLOT 90,140,1:DRAW95,145
1080 DRAW 84,145:DRAW 89,140
1082 PLOT 85,140,1:DRAW 95,140
1084 PLOT 140,120,1
1090 DRAW 169,120
1092 DRAW 169,140
1094 DRAW 139,140
1096 DRAW 140,120
1098 PLOT 90,130,1:DRAW110,130
1100 DRAW 110,134:DRAW 124,134
1101 GOTO 1116
1102 PRINT
1104 PRINTTAB(2);"ЦАП-1";TAB(20);"K"
1106 PRINTTAB(12);"VD"
1108 PRINT
1110 PRINTTAB(24);"Ф-294"
1112 PRINT
1114 PRINTTAB(2);"СП.";TAB(12);"R"
1115 GOTO 1050
1116 PRINT:PRINT:PRINT
1117 PRINT"      Схема  комутатора"
1118 PRINT"Уведіть значення опору резистора"
1120 INPUT"R=";R
1122 CLS
1124 RETURN
1140 PLOT X,Y,1
1142 DRAW X+A,Y
1144 DRAW X+A,Y+B
1146 DRAW X-1,Y+B
1148 DRAW X-1,Y
1150 RETURN
1152 PLOT X,Y,1
1153 DRAW X-A,Y
1154 PLOT X-2,Y+1,1
1155 PLOT X-2,Y-1,1
1156 PLOT X-4,Y+2,1
1157 PLOT X-4,Y-2,1
1158 RETURN
1160 PLOT X,Y,1
1161 DRAW X,Y+B
1162 PLOT X+1,Y+2,1
1163 PLOT X-1,Y+2,1
1164 PLOT X-2,Y+4,1
1166 PLOT X+2,Y+4,1
1170 RETURN
```

```
1180 PLOT X,Y,1
1181 DRAW X+A,Y
1182 PLOT X+2,Y+1,1
1184 PLOT X+2,Y-1,1
1186 PLOT X+4,Y+2,1
1188 PLOT X+4,Y-2,1
1190 RETURN
1192 POKE 73,149
1194 POKE 74,180
1196 X=USR(X)
1198 RETURN
1200 CLS
1201 PRINT"*********************************"
1202 PRINT"*                               *"
1204 PRINT"*                               *"
1206 PRINT"*                               *"
1208 PRINT"*  НАВЧАЛЬНА  ПРОГРАМА *"
1209 PRINT"*                               *"
1210 PRINT"*                               *"
1212 PRINT"*     <<EXPEREMENT>>      *"
1213 PRINT"*                               *"
1214 PRINT"*                               *"
1215 PRINT"*                               *"
1216 PRINT"*********************************"
1218 PRINT"натисніть клавішу <пропуск>"
1220 GOSUB 1192
1222 CLS
1224 PRINT"   Навчальна програма EXPEREMENT служить "
1226 PRINT" для моделювання основних функцій "
1228 PRINT" мікропроцесорної вимірювальної системи, "
1230 PRINT" на прикладі зняття ВАХ діода. "
1232 PRINT" Під управлінням програми реалізуються"
1234 PRINT слідуючі функції:"
1238 PRINT
1240 PRINT"  - уведення початкових умов;"
1242 PRINT"  - проведення вимірювань;"
1244 PRINT"  - обробка результатів вимірювань;"
1248 PRINT"  - виведення результатів вимірювань"
1250 PRINT"на зовнішній  пристрій у вигляді графіка."
1254 GOSUB 1192
1256 CLS
1260 PRINT"Структура вимірювальної системи:"
1261 PRINT
1262 PRINT TAB(12);"X2"
1266 PRINT TAB(6);"ПК-01";TAB(16);"5"
1267 PRINT
1268 PRINT
1269 PRINTTAB(8);"1       2    3    4"
1270 PRINT TAB(12);"X1"
1272 PRINT
1274 PRINTTAB(8);"6           7"
```

```
1276 PRINT
1280 X=33:Y=140:A=36:B=40
1282 GOSUB 1140
1284 X=88:Y=166:A=20:B=14
1286 GOSUB 1140
1288 X=69:Y=173:A=19
1290 GOSUB 1180
1292 X=88:Y=140:A=20:B=14
1294 GOSUB1140
1296 X=118:Y=140:A=20:B=14
1298 GOSUB 1140
1300 X=148:Y=140:A=20:B=14
1302 GOSUB 1140
1304 X=38:Y=113:A=20:B=14
1306 GOSUB 1140
1308 X=118:Y=113:A=20:B=14
1310 GOSUB 1140
1312 X=108:Y=173:A=50
1314 GOSUB 1180
1316 PLOT 158,173,1:DRAW 158,154
1318 X=88:Y=147:A=19
1320 GOSUB 1152
1322 X=118:Y=147:A=10
1324 GOSUB 1152
1326 X=148:Y=147:A=10
1328 GOSUB 1152
1330 X=48:Y=127:B=13
1332 GOSUB 1160
1334 X=123:Y=127:B=13
1336 GOSUB 1160
1338 X=133:Y=127:B=13
1340 GOSUB 1160
1342 PRINT"1 - керуюча ЕОМ;"
1344 PRINT"2 - інтерфейс;"
1346 PRINT"3 - цифро-аналоговий перетворювач (ЦАП);"
1350 PRINT"4 - вимірний комутатор;
1352 PRINT"5 - аналогово-цифровий перетвовач"
1353 PRINT" (вольтметр Ф-294)"
1354 PRINT"6 - дисплей;
1356 PRINT"7 - двокоординатний потенциометр (плотер)."
1360 GOSUB 1192
1362 RETURN
1400 CLS
1404 PRINT"Від'єднайте від ЦАП вимірний комутатор (гнізда X2, X3)."
1408 PRINT"Увімкніть плотер."
1410 GOSUB 1192
1412 CLS
1414 PRINT"Увага! Якщо незнаєте правил роботи з плотером "
1416 PRINT" зверніться до викладача, або лаборанта "
1420 GOSUB 1192
1422 CLS
```

```
1424 PRINT"Приєднайте до гнізда X2 входи плотера "
1426 PRINT" (вихід ЦАП-1 - вісь Y, вихід ЦАП-2 - вісь X),"
1428 PRINT" а до гнізда X3 - вхід управління пером "
1433 PRINT"Якщо готові - тисніть <проп.>"
1434 GOSUB 1192
1436 RETURN
```