НАВЧАЛЬНО-МЕТОДИЧНА ЛІТЕРАТУРА

Тернопільський національний технічний університет імені Івана Пулюя

Кафедра автоматизації технологічних процесів і виробництв

Електроніка і мікропроцесорна техніка

Методичні вказівки для лабораторної роботи № 25

Вивчення програмної моделі мікропроцесора. Програмування мікропроцесора

Тернопіль 2016

Методичні вказівки для лабораторної роботи №25. "Вивчення програмної моделі мікропроцесора. Програмування мікропроцесора" з курсу " Електроніка і мікропроцесорна техніка ". /Медвідь В.Р., Микулик П.М., Пісьціо В.П., Тернопіль: ТНТУ, 2016 - 10 с.

Для студентів напряму: 6.050202 "Автоматизоване управління технологічними процесами.

Методічні вказівки розглянуті і затверджені на засіданні кафедри автоматизації технологічних процесів і виробництв (протокол № 7 від 23.12.2015 року).

ЛАБОРАТОРНА РОБОТА №25
ВИВЧЕННЯ ПРОГРАМНОЇ МОДЕЛІ МІКРОПРОЦЕСОРА. ПРОГРАМУВАННЯ МІКРОПРОЦЕСОРА

Мета роботи

1. Вивчення способів запуску, ініціалізації та налаштування програмної моделі емуляції мікропроцесора.
2. Розгляд можливостей засобів для відображення роботи мікропроцесора.
3. Дослідження способів програмування і налагодження найпростішого мікропроцесора.
4. Знайомство з форматом команд та етапами їх виконання.
5. Вивчення команд пересилання даних і арифметичних команд.
6. Дослідження найпростіших програм.

Короткі теоретичні відомості

1. Програмна модель УМПК-80

Головне вікно програмної моделі УМПК-80 зі всіма активованими елементами меню «Перегляд» (“Просмотр”) представлене на рис. 1.

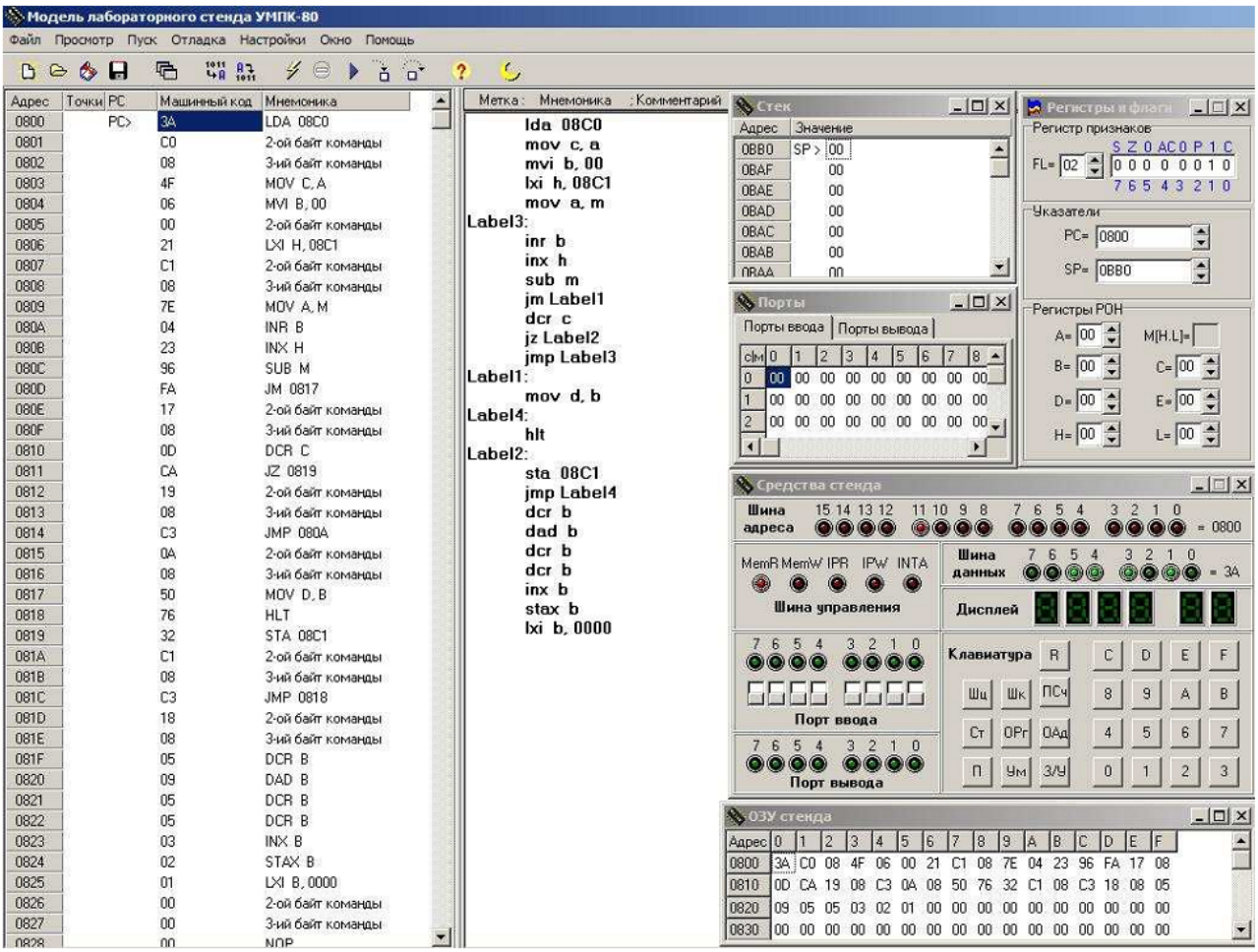

Рис. 1 Програмна модель УМПК-80

Програмна модель містить вичерпний посібник (див. меню «Допомога (“Помощь”)») і дозволяє працювати безпосередньо з адресами і їх вмістом (ліва частина рис. 1).

При цьому так само є можливість асемблювання і дизасемблювання програми користувача.

Допоміжні елементи «Стек», «Порти», «Засоби стенду», «ОЗП (ОЗУ) стенду»,

«Регістри і флаги» дозволяють більш гнучко і наочно проводити дослідження роботи мікропроцесора.

Двійкова система, як і десяткова, не є компактниою за формою, тому прийнято замість них використовувати **шістнадцяткову систему числення**.

Взаємозв'язок між двійковою **(BIN)**, десятковою **(DEC)** і шістнадцятковою **(HEX)** формою показана у вигляді табл. 1.

Таблиця 1. Системи числення

DEC	HEX	BIN	DEC	HEX	BIN
0	0	0000 0000 0000 0000	…	…	…
1	1	0000 0000 0000 0001	65519	FFEF	1111 1111 1110 1111
2	2	0000 0000 0000 0010	65520	FFF0	1111 1111 1111 0000
3	3	0000 0000 0000 0011	65521	FFF1	1111 1111 1111 0001
4	4	0000 0000 0000 0100	65522	FFF2	1111 1111 1111 0010
5	5	0000 0000 0000 0101	65523	FFF3	1111 1111 1111 0011
6	6	0000 0000 0000 0110	65524	FFF4	1111 1111 1111 0100
7	7	0000 0000 0000 0111	65525	FFF5	1111 1111 1111 0101
8	8	0000 0000 0000 1000	65526	FFF6	1111 1111 1111 0110
9	9	0000 0000 0000 1001	65527	FFF7	1111 1111 1111 0111
10	A	0000 0000 0000 1010	65528	FFF8	1111 1111 1111 1000
11	B	0000 0000 0000 1011	65529	FFF9	1111 1111 1111 1001
12	C	0000 0000 0000 1100	65530	FFFA	1111 1111 1111 1010
13	D	0000 0000 0000 1101	65531	FFFB	1111 1111 1111 1011
14	E	0000 0000 0000 1110	65532	FFFC	1111 1111 1111 1100
15	F	0000 0000 0000 1111	65533	FFFD	1111 1111 1111 1101
16	10	0000 0000 0001 0000	65534	FFFE	1111 1111 1111 1110
…	…	…	65535	FFFF	1111 1111 1111 1111

2. Команди асемблера мікропроцесора КР580ВМ80

Найбільш ефективно в мікропроцесорах виконуються програми, зіставлені на мові низького рівня - Асемблері.

Асемблер дозволяє в зручному для людини вигляді представляти команди (у вигляді мнемонічного запису), які є для мікропроцесора елементарними операціями (пересилання, вводу-виводу, арифметичних операцій і т.п.).

Програма, складена на Асемблері, надалі повинна бути переведена в машинний код для запису в мікропроцесор або за допомогою спеціальних **програм-компіляторів,** або вручну за допомогою знань правил запису і виконання операцій в МП-системах.

Необхідно відзначити, що будь-яка програма, написана на мові високого або низького рівня, може бути декодована і представлена у вигляді коду на Асемблері.

Це може бути використано при відлагодженні, пошуку помилок в роботі системи, налаштуванні взаємодії між різними апаратними засобами і т.п. операціях.

Команди в розглянутому типі мікропроцесора в залежності від місця, займаного в пам'яті, можуть бути **одно-, дво- і трибайтовими**.

Більшість команд є **однобайтовими** і містять в собі тільки **код операції**.

Наприклад:

- код команди «78h» має **мнемоніку** «MOV A, B» і означає пересилання вмісту регістра «B» в регістр «A»;
- код команди «81h» має мнемоніку «ADD C» і означає додавання вмісту регістра «C» до вмісту регістра «A», результат заноситься в регістр «A».

Необхідно відзначити що акумулятор (реєстр «A») у багатьох операціях явно не вказується, але мається на увазі.

У **двобайтовій** команді на першому місці, як і раніше, вказується **код** виконуваної операції (КОП), а в другому байті наводиться число, яке є **операндом** (якщо виконується

будь-яка операція), або номером пристрою вводу- виводу (при виконанні операції обміну даними).

Наприклад:

- код команди «06h A5h» має мнемоніку «MVIB, A5h» і означає пересилання константи в шістнадцятковій системі числення «A5h» в регістр «B».

Байти **трибайтової** команди мають таке призначення:

- перший байт - код операції;
- другий байт - молодші розряди двобайтового операнду;
- третій байт - старші розряди двобайтового операнду.

При цьому другий і третій байти використовуються для вказівки двобайтової адреси (команди або комірки пам'яті), або двобайтового операнду при роботі з парою регістрів.

Наприклад:

- код команди «3Ah 2Bh 09h» має мнемоніку «LDA 2bh, 09h» і означає пересилання в акумулятор вмісту комірки пам'яті (КП), адреса якої вказана як «092Bh».

Необхідно ще раз особливо підкреслити способи завдання адрес в КР580 - **спочатку вказується молодша частина адреси, потім старша.**

Нижче розглянуто деякі групи команд МП К580ВМ80, які необхідні для виконання роботи.

2.1. Команди пересилання даних

Команди пересилання даних здійснюють обмін даними між регістрами і пам'яттю, або між регістрами.

Основні команди цієї групи мають позначення **«MOV»** або **«MVI»**.

За командою «MOV» відбувається пересилання вмісту одного регістра в інший або вмісту регістра в пам'ять, або вмісту пам'яті в регістр.

Наприклад, по команді «MOV B, H» відбувається з передача числа з регістра «H» в регістр «Б».

За командою «MVI» здійснюється безпосередный запис числа в акумулятор, регістр або пам'ять (тобто саме число безпосередньо присутнэ в команді, сама команда двобайтова).

Наприклад, команда «MVI B, 35h» дозволяє безпосередньо записати число «35» (в шістнадцятковому коді) в регістр «B».

При записі команд мовою асемблера акумулятор позначається літерою «A» (в коді деяких команд явно не присутній). Константа, присутня в команді, позначається «d8».

Комірка пам'яті може позначатися як «M». При цьому використовується непряма адресація, тобто фактично звернення відбувається до комірки пам'яті, адреса якої записана в парі регістрів «H» (старша частина адреси) і «L» (молодша частина адреси).

2.2. Арифметичні команди

Команди цієї групи призначені для виконання операцій додавання, віднімання, збільшення або зменшення вмісту регістрів або комірок пам'яті на одиницю.

Команди додавання і віднімання завжди виконуються між першим операндом (числом), що знаходяться в акумуляторі, і другим операндом (числом, яке вказане в команді, знаходиться в регістрі або в пам'яті).

Результат виконання операцій завжди поміщається в акумулятор.

2.3. Приклад програми

Складемо програму для обчислення формули:

$$(X - 200 + Y) \rightarrow Z,$$

де «X» - вміст комірки пам'яті з адресою «0B00h»,

«Y» - вміст комірки пам'яті з адресою «0B01h»,

в яких розташовуються перший і другий операнд;

«200b» - задана константа в десятковій системі числення;

«Z» - вказує на позиціювання результату, а саме на комірку пам'яті з адресою «0B02h».

Відповідно до формули потрібно завантажити з адреси «0B00h» число «X», відняти від нього константу 200 і додати до числа «Y», завантаженого з комірки пам'яті з адресою «0B01h».

Результат необхідно зберегти в пам'яті за адресою «0B02h».

Вирішення задачі

Схему вирішення даної задачі можна представити у вигляді декількох етапів, розбитих відповідно до використовуваних в них типів операцій.

Етап 1. Здійснюється завантаження одного з регістрів вмістом комірки пам'яті з адресою «0B00h». Ця операція може бути виконана командою прямого завантаження акумулятора, мнемокод цієї команди — LDA adr. У нашому випадку рядок програми на Асемблері матиме вигляд

LDA 0B00h

Етап 2. Відбувається виконання операції віднімання: від вмісту акумулятора віднімається число «200», яке в шістнадцятковій системі обчислення записується як «C8h».

Мнемокод відповідної команди «SUI d8», результат поміститься в акумулятор. Для даних нашого завдання остаточно будемо мати

SUI C8h

Етап 3. Отриманий проміжний результат на етапі 2, збережений в акумуляторі, повинен бути доданий до вмісту комірки пам'яті «Y».

Пред цим необхідно завантажити пару регістрів «H, L» адресою комірки Y. Це здійснюється за допомогою команди «LXI H, d16» - безпосереднє завантаження пари регістрів «H», «L» зазначеним в команді двобайтовим операндом «D16». У нашому прикладі відповідний рядок на Асемблері запишеться наступним чином:

LXI H, 0B01h

Етап 4. Додамо до вмісту «A» вміст комірки пам'яті, що адресується парою регістрів H, L. Ця операція додавання виконується командою

ADD M.

Етап 5. Занести отриманий результат (який поки знаходиться в акумуляторі) в комірку пам'яті «Z» з адресою «0B02h». Мнемокод цієї команди - «STA adr». У нашому випадку маємо

STA 0B02h.

Для запису всієї програми в пам'ять мікропроцесора, як було відзначено раніше, необхідно перевести отриману програму в **машинні коди, де всі команди і числа представляються в шістнадцятковій системі числення.**

При цьому необхідно пам'ятати про розміри кожної з команд, про правила розміщення програм в пам'яті (викладених вище), і про машинні коди кожної з команд.

Тут і далі всі програми рекомендується представляти у вигляді таблиці 1, при цьому **умовно вважається за початок розташування програмного коду адреса «0800h».**

Така таблиця досить зручна і інформативна — крім мнемокодів команд на Асемблері і пояснень, вона містить все необхідне для програмування: адресу комірки пам'яті і те, що в неї необхідно записати.

Таблиця 1. Програма 1

№ етапу	Адреса, *h*	Мнемокод	Кількість байтів в команді	Машинний код, *h*	Коментар
1	0800	*LDA* 0B00*h*	3	3A	**Призначення команди *LDA*: вміст комірки пам'яті, адреса якої вказується в другому та третьому байтах команди, завантажити в регістр "A"**
	0801			00	**Молодший байт адреси**
	0802			0B	**Старший байт адреси**
2	0803	*SUI* C8*h*	2	D6	**Призначення команди *SUI*: від вмісту "A" віднімається другий байт команди**
	0804			C8	**Другий байт команди (операнд)**
3	0805	*LXI* H,0B01*h*	3	21	**Призначення команди *LXI*: безпосереднє завантаження пари регістрів "H" та "L"**
	0806			01	**Молодший байт адреси, який пересилається в регістр "L"**
	0807			0B	**Старший байт адреси, який пересилається в регістр "H"**
4	0808	*ADD* M	1	86	**Призначення команди *ADD*: вміст комірки пам'яті, адреса якої записана в регістрах "H", "L", додається до вмісту "A", результат - в "A"**
5	0809	*STA* 0B02*h*	3	32	**Призначення команди *STA*: вміст "A" пересилається в комірку пам'яті, адреса якої вказується в другому та третьому байтах команди**
	080A			02	**Молодший байт адреси**
	080B			0B	**Старший байт адреси**
-	080C	*HLT*	1	76	**Призначення команди *HLT*: Зупинка**

3. Виконується на самостійній підготовці

Завдання 1. Запуск, ініціалізація і налаштування засобів програмування і емуляції

1.1. Запустіть зазначену викладачем програму (включіть в роботу стенд).

1.2. Вивчіть загальний вигляд стенду, запустіть (у разі наявності) допомогу по засобу шляхом натискання на клавіатурі клавіші «F1» або вибравши відповідну позицію в меню.

1.3. Ознайомтеся з роботою програмного засобу (приведена в попередніх розділах) і основними правилами роботи з ним; вивчіть інформацію про мікропроцесор К580ВМ80.

1.4. Вивчіть методику налаштування і виконайте її для обраного засобу (каталогів програми, необхідних адрес, портів, переривань і іншими параметрами).

1.5. Занесіть в звіт короткі результати виконання пунктів даного завдання, забезпечивши їх малюнками (скріншот) і Вашими висновками.

Завдання 2. Дослідження засобів візуального відображення роботи мікропроцесора

2.1. Вивчіть можливості моделі по відображенню роботи мікропроцесора (наявні індикатори, засоби числового відображення, монітори, дисплеї, структурні і мнемонічні схеми); в разі необхідності активуйте додаткові вікна в програмі для відображення цих

функцій.

2.2. Розгляньте систему команд наявного мікропроцесора (активуйте додаткове вікно або розділ довідки, подивіться додатки методичного посібники), способи запису команд і даних в пам'ять МП-системи.

2.3. Занесіть в звіт інформацію і способи її аналізу за наступними ключовим елементам системи: шина адреси, шина даних, шина управління, комірки ОЗП, область пам'яті для стеку, флаги, регістри, елементи аналізу стану МП, буферні елементи, порти вводу-виводу.

4. Виконується в лабораторії

Порядок роботи

Завдання 1. Запис в пам’ять мікропроцесора програми.

Занести программу 1 (табл. 1) в пам'ять мікропроцесорної системи.

Записати за адресами «0B00h» и «0B01h» числа по своєму варіанту відповідно до табл. 2.

Таблиця 2. Варіанти чисел для Програми 1

Адреса комірок, *h*	**Значення чисел відповідно до варіантів, *h***														
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0B00	05	0E	5A	12	B5	B0	25	A0	EE	34	55	D5	12	6F	1D
0B01	06	0F	6A	11	B6	A0	26	99	FF	35	54	D6	13	7F	1E

Завдання 2. Дослідження результатів виконання програми в а**втоматичному режимі**

Здійснити запуск програми 1 в автоматичному режимі. Прочитати результат виконання програми з пам'яті (комірка пам'яті з адресою «0B02h») і занести його до звіту.

Порівняти отриманий практичний результат з розрахунками, зробленими вручну (перевести вихідні дані в двійкову систему числення і виконати відповідні перетворення).

Завдання 3. Дослідження виконання програми в **покроковому режимі**

Очистити вміст всіх регістрів системи і вміст комірки пам'яті з адресою «0B02h» (інші комірки ОЗП залишити без зміни).

Перевести систему в покроковий режим роботи (кожна команда виконується в кілька етапів). Результат роботи емулятора на кожному кроці виконання програми використовувати для заповнення табл. 3. Звірити результати табл. 3 з табл. 1.

Зробити висновок за отриманими результатами.

Примітка: Покроковий режим може бути або по-тактовий, або по-цикловий в залежності від можливостей емулятора.

Таблиця 3. РЕЗУЛЬТАТ виконання Програми 1 по тактах

Адреса, *h*	**Мнемокод**	**Кількість байтів в команді**	**Машинний код, *h***	**Кількість циклів (тактів) в команді**	**Біти регістра стану**								**Тип машинного циклу**
					***D*7**	***D*6**	***D*5**	***D*4**	***D*3**	***D*2**	***D*1**	***D*0**	
0800	*LDA* 0B00*h*	3	3A 00 0B	4 цикли									
0803	*SUI* C8*h*	2	D6 C8	2 цикли									
…	…	…	…	…	…								

Примітка: У разі відсутності можливості аналізу бітів регістра станів, відповідна колонка табл. 3 модифікується, і в ній вказують, наприклад, стан ліній шини управління.

Завдання 4. Дослідження модифікованої програми

Замінити в програмі 1 команду «SUI d8» на команду за варіантом (див. табл. 3).

Скласти для модифікованої програми таблицю, ідентичну табл. 2, занести її в звіт.

Для модифікованої програми виконати дослідження по завданнях 1 і 2, занести отримані дані в звіт.

Примітка: Слід пам'ятати, що початкова команда в програмі 1 з мнемокодом «SUI C8h» займала в пам'яті 2 байти. У зв'язку з цим, якщо модифікована команда містить меншу кількість байт, то початкову програму необхідно модифікувати (або скорегувати адреси розміщення програми, або скориставшись командою «NOP» - немає операції).

Таблиця 3. Варіанти чисел для Програми 1

Команда, на яку відбувається заміна "*SUI* d8" відповідно до номера варіанту														
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
ADI 01h	*ADI 1Ah*	*ADI 03h*	*ADI DBh*	*SBI DBh*	*SBI BBh*	*SBI 12h*	*SBI 5Fh*	*ACI 33h*	*ACI 81h*	*ACI F4h*	*ACI 0Ah*	*INR A*	*DCR A*	*DAA*

Занесіть в звіт короткі результати виконання пунктів даного завдання, забезпечивши їх малюнками (скріншот) і Вашими висновками.

Контрольні запитання

1. Дайте визначення основних використовуваних шин в МП-системі.
2. З чого складається узагальнена структура процесора?
3. Що таке акумулятор?
4. Що таке РЗП (РОН) і для чого вони призначені?
5. Наведіть основні флаги, що використовуються в регістрі ознак.
6. Що таке байт інформації?
7. Для чого потрібен покажчик стеку? Що таке стек?
8. Яким чином використовується дешифратор коду операцій?
9. Запишіть правила і приклади переходу від однієї системи числення до іншої.
10. Для чого потрібен програмний лічильник? Як він використовується?
11. Поясніть принцип роботи мікропроцесорної системи при виконанні програми користувача.
12. Для чого призначений програмний лічильник?
13. Які етапи створення завершених програм існують?
14. Що міститься в першому байті будь-якої команди?
15. Скільки всього байт може займати команда в пам'яті системи?
16. Що може міститися в другому і третьому байтах команди?
17. Охарактеризуйте команди пересилання даних (приклади, скільки байтів містять, специфіка виконання).
18. Охарактеризуйте арифметичні команди (приклади, скільки байтів містять, специфіка виконання).
19. Для чого призначений регістр стану системи?
20. Чим відрізняються машинні цикли від машинних тактів?

Список рекомендованої літератури

1. Самофалов К. Г., Викторов О. В. Микропроцессоры. – Б-ка инеженера – 2-е изд., перераб. и доп. - К: Техника, 1989.-312 с.
2. Шевкопляс Б.В. Микропроцессорные структуры. Инженерные решения: Справочник. 2-е изд., перераб. и доп. -М.: Радио и связь, 1990, -512с.
3. Угрюмов Е.П. Цифровая схемотехника. Спб.: BHV, 2001. 528 с.
4. Злобин В. К-, Григорьев В. Л. 58 Программирование арифметических операций в микропроцессорах: Учеб. пособие для технических вузов.— М.: Высш. шк., 1991. —303 с.: ил.