МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Запорізький національний технічний університет

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт з дисципліни
“Основи мікропроцесорної техніки”
для студентів всіх форм навчання спеціальностей
6.092206 “Електричні машини та апарати” зі спеціалізацією
“Електричні апарати” та
6.092204 “Електромеханічне обладнання енергоємних виробництв”

Частина 2

2010

Методичні вказівки до виконання лабораторних робіт з дисципліни “Основи мікропроцесорної техніки” для студентів всіх форм навчання спеціальностей 6.092206 “Електричні машини та апарати” зі спеціалізацією “Електричні апарати” та 6.092204 “Електромеханічне обладнання енергоємних виробництв”. Частина 2 / Укл.: Л.Б. Жорняк, В.І. Осинська. – Запоріжжя: ЗНТУ, 2010. –50 с.

Укладачі: Л.Б. Жорняк, доцент, к.т.н.
В.І. Осинська, асистент

Рецензент: М.О. Поляков, доцент, к.т.н.

Відповідальний
за випуск: О.В. Близняков, доцент, к.т.н.

Затверджено
на засіданні кафедри
«Електричні апарати»

Протокол № 11
від 16.06.2010 р.

ЗМІСТ

1 ЛАБОРАТОРНА РОБОТА № 6
ДВІЙКОВА АРИФМЕТИКА МІКРОПРОЦЕСОРА. ОПЕРАЦІЇ І КОМАНДИ.

1.1 Мета роботи

Вивчення арифметичних операцій і команд мікропроцесора КР580ВМ80А.

1.2 Загальні відомості

1.2.1 Незалежно від форми подання чисел у вихідних програмах МП оперує тільки з двійковими числами, що має свою специфіку.

Поняттям двійкова арифметика відображується специфіка арифметичних операцій із двійковими числами. Базовими арифметичними операціями МП є додавання і віднімання двійкових чисел, що випливає з принципів організації обчислювальних процесів.

Додавання двійкових чисел, як і десяткових, виконується за розрядами від молодшого до старшого, але значно простіше в реалізації, тому що числа мають тільки два значення. Найбільшу відповідність природному представленню додавання дає таблична форма, у якій додатки утворюють рядки і стовпці, а результат додавання цифр одноіменних розрядів знаходиться на перетинанні відповідних рядка і стовпця. Стосовно до додавання двійкових цифр зазначена таблиця має вид табл. 1.1.

Таблиця 1.1 - Додавання двійкових чисел

Додаток	Додаток		
	+	0	1
	0	0	1*
	1	1	0

* має місце перенос одиниці в старший розряд.

Додавання двійкових чисел виконується за наступними правилами:

- додавання двох одиниць дає нуль у молодшому розряді результату і перенос одиниці в старший розряд (1+1=0*);

- додавання одиниці до нуля молодшого розряду дасть у результаті одиницю без переносу в старший розряд (1+0=1);

- нулі, що складаються, результатом дають нуль (0+0=0).

При всій простоті двійкового додавання як процедури, громіздкість запису великих чисел у двійковій формі викликає необхідність безлічі переносів з одного розряду в інший, що створює певні незручності.

Двійкове віднімання також багато в чому подібне десятковому. Таблична форма запису в даному випадку має вид табл. 1.2.

Таблиця 1.2 - Віднімання двійкових чисел

Зменшуване	Від'ємне		
		0	1
0	0	0	1*
1	1	1	0

* означає позику одиниці із сусіднього старшого розряду.

Основні правила віднімання:

- віднімання з одиниці одиниці і з нуля нуль дає нуль (1 - 1 = 0, 0 - 0 = 0);

- віднімання з одиниці нуля дає одиницю (1 - 0 = 1);

- віднімання з нуля одиниці вимагає позики одиниці зі старшого розряду і дає в результаті одиницю (0 - 1 = 1).

До числа елементарних арифметичних операцій відносять зсув двійкового числа на один розряд уліво чи управо, порівняння двох двійкових чисел, а також збільшення чи зменшення числа на 1.

До числа складних арифметичних операцій відносять такі, які можна представити у виді комбінації декількох простих. Так множення можна представити у виді комбінації додавання і зсуву, а ділення у виді послідовності віднімання і зсуву.

У мікропроцесорній системі КР580, варіантом якої є НМК, передбачена реалізація таких команд двійкової арифметики:

а) додавання восьмирозрядних чисел;

б) додавання шістнадцятирозрядних чисел;

в) віднімання восьмирозрядних чисел;

г) інкремент;

д) декремент.

Всі арифметичні операції з восьмирозрядними операндами в МП КР580 основані на уявлені про те, що один з операндів розміщується в акумуляторі (регістр А), а інший - у регістрі (РЗП) чи в пам'яті (М). Адреса комірки пам'яті (М) вказується в регістровій парі HL. Другий операнд може бути також числом, заданим безпосередньо в самій команді. Результат арифметичної операції записується в акумулятор. Операція віднімання має ту особливість, що робиться завжди із вмісту акумулятора, тобто зменшуване повинне бути операндом, розміщеним в акумуляторі. По результаті арифметичних операцій додавання і віднімання встановлюються такі типи ознак:

С - переносу, Z - нуля, S - знаку, P - парності, AC - допоміжного переносу.

Додавання шістнадцятирозрядних двійкових чисел у зазначеній системі називають подвійним додаванням. Воно засновано на тому, що один з операндів знаходиться в регістровій парі HL, а другий - або в DE, або у BC. Результат записується в HL. По результату операції встановлюється чи скидається біт переносу - С.

Операція інкремента полягає в збільшенні вмісту регістрів, регістрових пар, комірки пам'яті за адресою в HL на одиницю. Інкремент регістра і пам'яті змінює біти ознак: Z, S, P, AC. Інкремент регістрової пари не торкається бітів ознак.

Операція декремента полягає в зменшенні вмісту регістрів, регістрових пар, комірки пам'яті за адресою в HL на одиницю. Змінювані біти ознак аналогічні команді інкремента.

1.2.2 Команди додавання восьмирозрядних чисел:

- ADD R - додавання регістра: A, B, C, D, E, H, L;
- ADD M - додавання комірки пам'яті (адреса в HL);
- ADI B - додавання безпосереднього числа B;
- ADC R - додавання регістра:A,B,C,D,E,H,L плюс біт переносу С;
- ADC M - додавання комірки пам'яті (адреса в HL) плюс біт переносу С;
- ACI B - додавання безпосереднього числа B плюс біт переносу С.

Як приклад, виконайте таке завдання:

- уведіть програму, приведену в табл. 1.3., що реалізує операцію A = A + B + M + 1;

Таблиця 1.3

Адреса	Команда	Код	Коментар
800	ADD B	80	A=A+B
801	LXI H, 900H	21 00 09	Завантаження адреси HL=900H
804	ADD M	86	A=A+M
805	ADI 01	C6 01	A=A+1

- виконайте програму, попередньо задавши вихідні значення відповідно до табл. 1.4 (M=900H): <СТ> 800 <–> 807 <ВП>;

- перевірте отримані результати (результат операції в регістрі A та біти умов у регістрі F).

Таблиця 1.4

Регістр								
	Початкові дані	A	00	00	00	F0	FF	55
		B	00	02	10	0E	00	AA
		M	00	03	45	00	00	FF
	Результат	A	01	06	56	FF	00	FF
		F	02	06	06	86	57	86

Як приклад, виконайте таке завдання:

- уведіть в пам'ять програми додавання шістнадцятирозрядних чисел на основі команд восьмирозрядного додавання представлену в табл. 1.5 для виразу: HL=DE+BC;

- виконайте програму, попередньо задавши вихідні значення відповідно до табл. 1.6, за допомогою такої дії: <СТ > 800 <–> 806 <ВП>;

- перевірте отримані результати в регістрах H, L та F .

Таблиця 1.5

Адреса	Команда	Код	Коментар
800	MOV A, C	79	A ß C
801	ADD E	83	A + E
802	MOV L, A	6F	L , ß A
803	MOV A, B	78	A ß B
804	ADC D	8A	D + B, з урахуванням переносу
805	MOV H, A	67	H ß A

Таблиця 1.6

Регістр							
	BC	0001	02C5	F000	8137	809F	FFFF
	DE	00FE	03FI	0FFF	72D9	8121	0001
	HL	00FF	06B6	FFFF	F410	01C0	0000
	F	46	06	86	82	03	57

1.2.3 Команди віднімання восьмирозрядних чисел мають вид:

- SUB R - віднімання регістра: A, B, C, D, E, H, L;
- SUB M- віднімання комірки пам'яті (адреса в HL);
- SUI B - віднімання безпосереднього числа B;
- SBB R - віднімання регістра: A,B,C,D,E,H,L мінус біт переносу C;
- SBB M - віднімання комірки пам'яті(адреса в HL), мінус біт переносу C;
- SBI B - віднімання безпосереднього числа мінус біт переносу.

Як приклад, виконайте таке завдання:

- уведіть в пам'ять програму, представлену в табл. 1.7., що реалізує вираз: A = A – B – M – 1;

Таблиця 1.7

Адреса	Команда	Код	Коментар
800	SUB B	90	A = A - B
801	LXI H, 900H	21 00 09	Завантаження HL =900H, адреса M
804	SUB M	96	A=A - M
805	SBI 1	DE 01	A=A - 1

- виконайте програму, попередньо задавши вихідні дані відповідно до табл. 1.8, командою: <СТ> 800 <–> 807 <ВП>;

Таблиця 1. 8

A	FF	00	01	25	00	05
B	01	FF	01	20	00	06
M	01	00	00	04	00	FF
A рез.	FC	00	FF	00	FF	FF
F	96	56	87	56	87	87

- перевірте отримані результати.

Як приклад, виконайте таке завдання:

- уведіть в пам'ять програму, приведену в табл. 1.9, віднімання шістнадцятирозрядних чисел: HL=DE-BC;

Таблиця 1.9

Адреса	Команда	Код	Коментар
800	MOV A, E	7B	E ß A
801	SUB C	91	E – C
802	MOV L, A	6F	L ß A
803	MOV A, D	7A	A ß D
804	SBB B	98	D – B з урахуванням переносу
805	MOV H, A	67	H ß A

- виконайте програму, попередньо задаючи вихідні значення відповідно до табл. 1.10: <СТ> 800 <–> 806 <ВП>;

Таблиця 1.10

Регістр	DE	0000	FFFF	0F0F	8000
	BC	FFFF	0F0F	0001	7FFF
	HL	0001	F00F	0FEF	0001
	F	47	96	16	46

- перевірте отримані результати.

1.2.4 Команди подвійного додавання мають вид:

- DAD H – додавання HL=HL+HL;
- DAD B – додавання HL=HL+BC;
- DAD D – додавання HL=HL+DE.

Як приклад, виконайте таке завдання:

- уведіть згідно з табл. 1.11 програму, що реалізує операцію: HL=BC+DE;

Таблиця 1.11

Адреса	Команда	Код	Коментар
800	MOV H, B	60	H ß B
801	MOV L, C	69	L ß C
802	DAD D	19	HL = HL + DE

- виконайте програму, задавши вихідні значення відповідно до табл. 1.12: <СТ> 800 <–> 803 <ВП>;

Таблиця 1.12

Регістр	BC	0000	7FFF	8000	55AA	ECB9	FFFF
	DE	FFFF	8000	8000	AA55	1347	8000
	HL	7FFF	FFFF	0000	FFFF	0000	7FFF
	F	46	46	47	46	47	47

- перевірте результати.

1.2.5 Команди інкремента мають вид:

- INR R - збільшення на одиницю вмісту регістрів: A, B, C, D, E, H, L;
- INR M - збільшення на одиницю вмісту комірки пам'яті, адреса M в HL;
- INX RP - збільшення на одиницю вмісту пари регістрів: BC, DE, SP. У даній команді вказується ідентифікатор старшого регістра (INX B).

Як приклад, виконайте таке завдання:

- запишіть в пам'ять команду INR E (E=E+1) за адресою 800H: 800 1C;

- виконайте зазначену команду для заданих значень вмісту регістра E згідно з табл. 6.13: <СТ> 800 <–> 801 <ВП>;

Таблиця 1.13

Регістр	E поч.	00	0F	F0	FF
	E рез.	01	10	F1	00
	F	02	12	82	56

- перевірте отриманий результат.

Завдання:
- запишіть в пам'ять коди команд:
800 21 00 09 LXI H, 900H Завантаження. HL = 900H, адреса M
803 34 INR M M = M + 1
- виконайте зазначену послідовність для заданих в табл. 6.14 значень вмісту комірки пам'яті M: <СТ> 800 <–> 804 <ВП>;

Таблиця 1.14

Регістр	M поч.	00	0F	F0	FF
	M рез.	01	10	F1	00
	F	03	13	83	57

- перевірте отримані результати.

Завдання:
- запишіть в пам'ять код команди
800 13 INX D DE = DE + 1
- виконайте приведену команду для заданих у табл. 1.15 значень вмісту регістрової пари DE: <СТ> 800 <–> 801 <ВП>;

Таблиця 1.15

DE поч.	0000	OFOF	OOFF	FFFF
DE рез.	0001	0F10	0100	0000
F	56	56	56	56

- перевірте отримані результати.

Примітка: Значення регістра ознак F залишається рівним останньому значенню попереднього завдання через те, що інкремент пари регістрів не змінює ознак.

1.2.6 Команди декремента мають вид:

- DCR R - зменшення на одиницю вмісту регістра: A, B, C, D, E, H, L;
- DCR M - зменшення на одиницю вмісту комірки пам'яті - адреса M у HL;
- DCX RP зменшення на одиницю вмісту пари регістрів: BC, DE, HL, SP.

У команді вказують ідентифікатор старшого регістра, наприклад, DCX B.

Як приклад, виконайте таке завдання:

- запишіть в пам'ять таку команду:

800 0D DCR C C=C - 1.

- виконайте зазначену команду для заданих у табл. 6.16 значень регістра C: <СТ> 800 <–> 801 <ВП>;

Таблиця 1.16

C поч.	00	01	10	11
C рез.	FF	00	0F	10
F	87	57	07	13

- перевірте отримані результати.

Завдання:

- запишіть в пам'ять коди команд:

800 21 00 09 LXI H, 900H Завантаження HL = 900H, адреса M;
803 35 DCR M: M = M – 1.

- виконайте зазначені команди для початкових значень вмісту комірки пам'яті M, заданих у табл. 1.17;
- виконайте команда запуску: <СТ> 800 <–> 804 <ВП>;
- перевірте отримані результати.

Таблиця 1.17

M поч.	00	01	10	11
M рез.	FF	00	0F	10
F	87	57	07	13

Завдання:

- запишіть в пам'ять код команди:

800 2B DCX H HL = HL - 1

- виконайте команду для значень вмісту регістрової пари H, заданих у табл. 1.18.

Таблиця 1.18

H поч.	0000	1000	FFFF	0001
H рез.	FFFF	0FFF	FFFE	0000
F	13	13	13	13

- команда запуску: <СТ> 800 <–> 801 <ВП>;
- перевірте отримані результати.

Примітка: Команда декремент регістрової пари не торкається вмісту бітів ознак.

1.3 Порядок виконання роботи

1.3.1 Написати і виконати програму, що реалізує операцію: C=D+E, дані використати з табл 6.19

Таблиця 1.19

D	10	FF	C7	19	AA	E5
E	80	01	08	49	55	F0

Скласти таблицю результатів C і F.

1.3.2 Написати і виконати програму додавання вмісту двох комірок пам'яті: Ml = M2 + M3 з використанням адрес Ml = 900H, M2 = 901 H, M3 = 902 H.

Вміст комірок пам'яті (додатків) наведений в табл. 1.20.

Таблиця 1.20

M2	00	FE	D5	22	61	19
M3	F0	02	C2	BB	95	33

Скласти таблицю результатів M1, F.

1.3.3 Написати і виконати програму додавання:
HL = BC + DE+ 4000H.

Початкові дані взяти довільно, у таблиці результатів показати D, E, C, F.

1.3.4 Написати програму віднімання: C = D - E - 10H.

Початкові дані взяти довільно, у таблиці результатів показати D, E, C, F.

1.3.5 Написати програму віднімання вмісту двох комірок пам'яті: Ml = M2 – M3.

Адреси комірок пам'яті Ml = 900H, M2 =901H, M3 = 902H. Заповнити таблицю, у якій відбити вміст M2, M3, Ml та F.

1.3.6 Написати і виконати програму віднімання: HL = BC - DE - 0FFFH для довільних початкових даних. Заповнити таблицю зі значеннями BC, DE, HL, F.

1.3.7 Написати і виконати програму заповнення масиву по заданому індексу елемента масиву відповідно до табл. 6.21.

Таблиця 1.21

Базова адреса масиву	900					
Номер елемента	00	03	05	11	18	1F
Адреса елемента						
Вміст	00	01	02	03	04	05

Перевірте правильність роботи програми.

1.3.8 Написати і виконати програму перезапису вмісту масиву 1 за завданням 1.3.7, у масив 2 відповідно до табл. 1.22.

Таблиця 1.22

Базова адреса масиву	910
Номер елемента	2 4 6 8 A C
Адреса елемента	
Вміст	

Перевірте правильність роботи програми.

1.3.9 Написати і виконати програму заповнення масиву пам'яті (900H - 904H) даними (00 - 04), використовуючи команди інкремента регістра і пари регістрів.

1.3.10 Написати і виконати програму заповнення масиву пам'яті (90FH - 90AH) даними (OFH - OAH), використовуючи команди декремента регістра і пари регістрів.

1.4 Зміст звіту

Звіт повинен містити:

– назву та мету роботи;

– написані програми відповідно до завдань 1.3;

– результати виконання програм;

– висновки.

1.5 Контрольні запитання

1.5.1 Характеристика елементарних операцій двійкової арифметики.

1.5.2 Особливості і форми представлення операцій додавання і віднімання двійкових чисел.

1.5.3 Сутність і призначення операцій подвійного додавання, віднімання.

1.5.4 Операції зсуву, їхнє практичне значення.

1.5.5 Принципи заповнення і переносу масивів.

2 ЛАБОРАТОРНА РОБОТА № 7
ЛОГІЧНІ ОПЕРАЦІЇ. ОРГАНІЗАЦІЯ І КОМАНДИ.

2.1 Мета роботи

Вивчення принципів, організації і команд логічних операцій.

2.2 Загальні відомості

Багато застосувань мікропроцесорних пристроїв припускають логічну обробку інформації . Тому вони здатні замінити собою безліч логічних схем , реалізувати складні логічні функції на основі використання елементарних логічних операцій.

У системі команд МП КР580 для реалізації логічних операцій існують такі команди :

– логічне додавання;
– логічне множення;
– вилучаюче АБО;
– інверсія.

Команди логічного додавання реалізують логічну операцію АБО. Результат дорівнює 1 , якщо хоча б один з відповідних байтів дорівнює 1, і дорівнює нулю, якщо обоє дорівнюють нулю. Наприклад:

10101001
OR
00110010
10111011

де OR - позначення операції АБО .

Команди логічного множення реалізують логічну операцію І. Результат дорівнює 1, якщо обоє відповідних біта рівні 1, і дорівнює нулю , якщо хоча б один з них дорівнює нулю. Наприклад :

10101000
AND
00110010
00100000,

де AND - позначення операції логічне І.

Команди вилучаючого АБО реалізують логічну операцію ДОДАВАННЯ ПО МОДУЛЮ ДВА . Результат дорівнює 1 , якщо відповідні біти протилежні (1 і 0) , дорівнює 0 , якщо вони однакові (1 і 1, 0 і 0) . Наприклад :

10101001
XOR
00110010
10011011

де XOR - позначення операції ДОДАВАННЯ ПО МОДУЛЮ ДВА .

Команда інверент реалізує операцію ЗАПЕРЕЧЕННЯ вмісту акумулятора . Наприклад :

10101001
NOT_____
01010110

де NOT - позначення операції ЗАПЕРЕЧЕННЯ .

Усі логічні команди виконуються побітно з восьми розрядними операндами . При цьому один з операндів розміщується в регістрі накопичувачі, акумуляторі , а другий - або в одному з регістрів загального призначення , або в комірці пам'яті , або задається в другому байті команди . Результат виконання команди записується в акумулятор. При цьому біт переносу встановлюється в нуль, а інші біти установлюються відповідно до результату виконання команди .

2.2.1 Команди логічного додавання

- ORA R - з регістром: A, B, C, D, E, H, L;
- ORA M - з коміркою пам'яті , адреса якої в HL;
- ORI B - з безпосереднім операндом B.

Як приклад, рекомендується виконати такі завдання:

а) записати в пам'ять коди програми , відповідно до табл. 2.1, що реалізує вираз : A = (A OR C OR M OR 80H).

Таблиця 2.1

Адреса	Команда	Код	Коментар
800	ORA C	B1	A = A OR C
801	LXI H, 900H	21 00 09	Завантажити HL = 900 H, адреса M
804	ORA M	B6	A = A OR M
805	ORI 80H	F6 80	A = A OR 80H

Задаючи вихідні значення відповідно до табл. 2.2., виконати програму командою <СТ> 800 <–> 807 <ВП>.

Таблиця 2.2

A вих.	00	FF	AA	01	70	DD
C	00	00	55	10	04	BB
M (900H)	00	00	88	02	53	CC
A рез.	80	FF	FF	93	F7	FF
F	82	86	86	86	82	86

Перевірити отримані результати.

б) записати в пам'ять коди програми відповідно до табл. 2.3, що реалізує вираз HL = (BC OR DE).

Таблиця 2.3.

Адреса	Команда	Код	Коментар
800	MOV A, C	79	Пересилання A ← C
801	ORA E	B3	A = A OR E
802	MOV L, A	6F	Пересилання L ← A мол. байта результату
803	MOV A, B	78	Пересилання A ← B
804	ORA D	B2	A = A OR D
805	MOV H, A	67	Пересилання H ← A, ст. байта результату

Попередньо задаючи вихідні значення відповідно до табл. 2.4, виконати програму командою: <СТ> 800 <–> 806 <ВП>.

Таблиця 2.4

BC	0000	AA55	0123	FD03
DE	FFFF	55AA	4567	1057
HL	FFFF	FFFF	4567	FD57
F	86	86	02	82

Перевірити отримані результати.

2.2.2 Команди логічного множення:

- ANA R - з регістром A, B, C, D, E, H, L;
- ANA M - - з коміркою пам'яті, адреса якої в HL;
- ANA B - з безпосереднім операндом B.

Як приклад, рекомендується виконати такі завдання:

а) записати в пам'ять з табл. 2.5 коди програми, реалізуючої вираз: A = (A AND D AND M AND 7FH).

Таблиця 2.5

Адреса	Команда	Код	Коментар
800	ANA D	A2	A = A AND D
801	LXI H, 900H	21 00 09	Завантажити HL = 900 H, адреса M
804	ANA M	A6	A = A ANA M
805	ANI 7FH	F6 7F	A = A AND 7FH

Виконати програму, попередньо задаючи вихідні значення з табл. 2.6, командою <СТ> 800 <–> 807 <ВП> .

Перевірити отримані результати.

Таблиця 2.6

A вих.	00	FF	AA	55	81	DB
D	A2	FF	55	90	77	23
M (900H)	36	FF	FF	C3	6E	0F
A рез.	00	7F	00	00	00	03
F	56	12	56	56	56	16

б) записати в пам'ять коди програми згідно з табл. 2.7, що реалізує вираз HL = (B ORA C) AND DE AND FOFFH.

Таблиця 2.7

Адреса	Команда	Код	Коментар
800	MOV A, C	79	Пересилання A ← C
801	ORA B	B0	A = A OR B
802	ANA E	A3	A = A AND E
803	ANI FFH	E6 FF	A = A AND FFH
805	MOV L, A	6F	Пересилання L ← A, мол. байт результату
806	MOV A, D	7A	Пересилання A ← D
807	ANI F0H	E6 F0	A = A AND F0H
809	MOV H, A	67	Пересилання H ← A, ст. байт результату

Виконати програму, попередньо задаючи вихідні значення з табл. 2.8 та використовуючи команду <СТ> 800 <–> 80A <ВП>.

Таблиця 2.8

B	00	FF	C7	D0
C	01	23	50	CD
DE	1023	59ED	3579	ABCD
HL	1001	50ED	3051	A0CD
F	02	06	06	96

2.2.3 Команди додавання за модулем два такі:

- XRAR – з регістром: A, B, C, D, E, H, L;
- XRAM - з коміркою пам'яті, адреса якої в HL;
- XRIB - з безпосереднім операндом B.

Як приклад, рекомендується виконати такі завдання:

а) записати в пам'ять коди програми згідно з табл. 2.9, що реалізує вираз A = A XOR A XOR E XOR M XOR AAH .

Таблиця 2.9

Адреса	Команда	Код	Коментар
800	XRA A	AF	A = A XOR A, A = 0
801	XRA E	AB	A = A XOR E
802	LXI H, 900H	21 00 09	Завантажити HL =900 H, адреса M
805	XRA M	AE	A = A XOR M
806	XRI AAH	EE AA	A= A XOR AAH, рез.

Виконати програму, задаючи вихідні значення з табл. 2.10, командою <СТ> 800 <–> 808 <ВП> .

Таблиця 2.10

E	00	FF	C2	AA	00	33
M	F0	0F	2	55	00	CC
A	5A	5A	45	55	AA	55
F	06	06	02	06	86	06

Перевірити отримані результати.

б) записати в пам'ять відповідно до табл. 2.11 коди програми, реалізуючої вираз HL = (A OR B) AND DE XOR (HL XOR C).

Таблиця 2.11

Адреса	Команда	Код	Коментар
800	ORA B	B0	A OR B = A
801	ANA E	AE	A = A AND E
802	MOV E, A	5F	Пересилання E ← A, запис проміжного результату
803	MOV A, C	79	Пересилання A ← C
804	XRA L	AD	A = A XOR L
805	XRA E	AB	A = A XOR E
806	MOV L, A	6F	Пересилання L ← A, мол. байт результату
807	MOV A, H	7C	Пересилання A ← H, ст. байт результату
808	XRA D	AA	A = A XOR D
809	MOV H, A	67	Пересилання H ← A, ст. байт результату

Виконати програму, задаючи вихідні значення відповідно до табл. 2.12, командою <СТ> 800 <–> 80A <ВП> .

Перевірити отримані результати.

Таблиця 2.12

A	05	73	21	DE
B	66	2F	EB	32
DE	67 12	36 CD	BA FF	F2 35
HL вих.	33 55	79 21	68 AC	DB FE
C	A2	36	78	CD
HL рез.	54 A5	4F 5A	D2 3F	29 07
F	02	02	86	02

2.2.4 Команда інверсії така:

- CMA - інверсія акумулятора.

Як приклад, рекомендується виконати такі завдання:

а) записати в пам'ять програму, приведену в табл. 2.13, що реалізує вираз A = NOT ((NOT B) AND (NOT C)).

Таблиця 2. 13

Адреса	Команда	Код	Коментар
800	MOV A, B	78	Пересилання A ←B
801	CMA	2F	F = NOT A
802	MOV B, A	47	Пересилання B ← A (NOT B)
803	MOV A, C	79	Пересилання A ← C
804	CMA	2F	A = NOT A (NOT C)
805	ANA B	A0	A = A AND B
806	CMA	2F	A = NOT A, результат

Виконати програму, задаючи вихідні значення відповідно до табл. 2.14, командою <СТ> 800 <–> 807 <ВП> .

Таблиця 2.14

B	00	FF	25	39	AB	ED
C	FF	02	56	21	13	C3
A	FF	FF	77	39	BB	EF
F	56	56	96	96	16	12

Перевірити отримані результати.

б) записати в пам'ять програму відповідно до табл. 2.15, що реалізує вираз M3 = NOT (NOT Ml) OR (NOT M2) .

Адреси комірок пам'яті: Ml =900H, M2 = 901H, M3 = 902H.

Таблиця 2.15

Адреса	Команда	Код	Коментар
800	LXI H, 900 H	21 00 09	Завантажити HL = 900 H, адреса M1
803	MOV A, M	7E	Читання A = M1
804	CMA	2F	A= NOT A (NOT M1)
805	MOV C, A	4F	Пересилання C ← A, проміжний результат
806	INX H	23	HL = HL + 1, адреса M2 (901)
807	MOV A, M	7E	Читання A = M2
808	CMA	2F	A = NOT A (NOT M2)
809	ORA C	B1	A = A OR C
80A	CMA	2F	A = NOT A, результат
80B	INX H	23	HL = HL + 1, адреса M3 (902)
80C	MOV M, A	77	Запис M3 = A, результат

Виконати програму, задаючи вихідні значення відповідно до табл. 2.16, командою <СТ> 800 <–> 80D <ВП> .

Таблиця 2.16

M1 (900 H)	37	43	97	78
M2 (901 H)	29	5E	F6	95
M3 (902 H)	21	42	94	10
F	86	86	02	82

Перевірити отримані результати.

2.3 Порядок виконання роботи

2.3.1 Написати і виконати програму для реалізації виразу HL = (B OR C OR DE) OR 8800H .

Заповнити таблицю для значеньB, C, DE, HL, F Вихідні значення установити довільно.

2.3.2 Виконати те саме для виразу: M3 = Ml OR M2

Адреси комірок пам'яті M1 = 900H, M2 = 901H, M3 = 902H.Вихідні дані M1 і M2 узяти довільно. Результати представити в таблиці: M1, M2, M3, F.

2.3.3 Написати і виконати програму для реалізації виразу HL = (HL AND BC OR DE) AND 03FFH .

Вихідні значення взяти довільно. Заповнити таблицю: HL(поч.), BC, DE, HL(рез.), F .

2.2.4 Реалізувати вираз: M2 =A AND Ml OR C AND D.

Адреси комірок пам'яті: M1= 900H, M2 = 901H. Вихідні значення взяти довільно. У таблиці відобразити значення (вміст) A, C, D, Ml, M2, F .

2.3.5 Реалізувати вираз:

H = ((L XOR D) OR E AND H) XOR (B OR C) XOR 0FH.

Вихідні значення взяти довільно. У таблиці відобразити вміст регістрів : HL(поч.), D, E, B, C, HL (рез.), F .

2.3.6 Реалізувати вираз:

M = (M XOR AAH) AND (B OR E).

Адреса M = 900H. Вихідні значення взяти довільно. Скласти таблицю: M (поч.), B, E, M (рез.), F.

2.3.7 Реалізувати вираз:

HL = ((NOT DE) XOR (NOT BC)) AND (NOT(C OR (NOT B))).

Вихідні значення взяти довільно. Скласти таблицю : BC, DE, HL, F.

2.3.8 Реалізувати вираз:

M6 = (NOT M1) XOR M2 OR (NOT M3) AND (NOT(M4 OR M5))

Адреси комірок пам'яті : M1 = 900H, M2 = 901H, M3 = 902H, M4 = 903H, M5 = 904H, M6 = 905H.

Вихідні значення взяти довільно. Скласти таблицю: M1, M2, M3, M4, M5, M6, F.

2.4 Зміст звіту

Звіт повинен містити:
- назву та мету роботи;
- написані програми відповідно до завдань підрозділа 7.3 в мнемокодах та машиних кодах з коментарями;
- висновки.

2.5 Контрольні запитання

2.5.1 Особливості логічних операцій.
2.5.2 Сутність елементарних логічних операцій.
2.5.3 Технологія і команди логічного додавання.
2.5.4 Технологія і команди логічного множення.
2.5.5 Особливості і команди додавання за модулем два.
2.5.6 Зміст і практичне значення інверсії.

3 ЛАБОРАТОРНА РОБОТА № 8
ОПЕРАЦІЇ І КОМАНДИ ЗСУВУ

3.1 Мета роботи

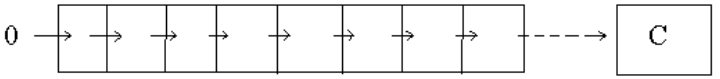
Вивчення операцій і команд простого та циклічного зсуву.

3.2 Загальні відомості

Команди зсуву дозволяють виконати на основі елементарних арифметичних операцій необхідний набір дій над числами. Крім того реалізують закінчені операції по обробці даних. Використовуються переважно при тестуванні та порівнянні бітів. Команди зсуву МП КР580 (Intel 8080) оперують тільки даними акумулятора і не вимагають інших операндів, розташованих у пам'яті чи регістрах. Спосіб адресації в даному випадку називають неявним, а часто просто не вказують. Багато які МП мають інші, ніж типовий МП типи команд зсуву. Наприклад, МП Motorola 68000 здійснює зсув вмісту не тільки акумулятора, але і комірки пам'яті, а команди операцій зсуву

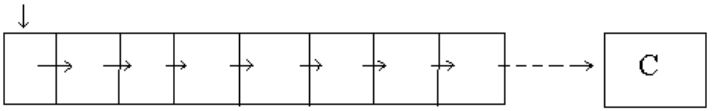
впливають не тільки на ознаку переносу, як це обстоїть у базових МП, але і на всі інші ознаки. Розглянемо діаграми можливих видів зсуву:

а) зсув управо відбувається таким чином:



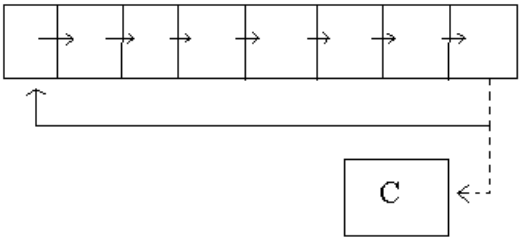
Послідовність дій при цьому така:
- крайній правий біт засилається в С (регістр переносу);
- всі інші біти зсуваються управо;
- у крайній лівий розряд засилається 0.

б) арифметичний зсув управо відбувається таким чином:



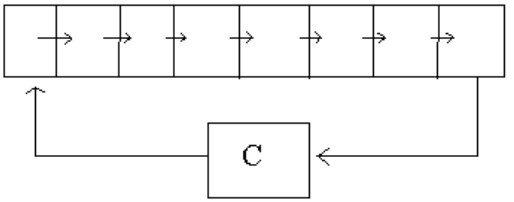
Послідовність дій при цьому така:
- крайній правий біт засилається в С;
- інші біти зсуваються управо;
- крайній лівий біт залишається без змін.

в) циклічний зсув управо відбувається таким чином:



Послідовність дій при цьому така:
- крайній правий біт засилається в С;
- інші біти зсуваються управо;
- у крайній лівий розряд засилається первісний вміст крайнього правого розряду.

г) циклічний зсув управо з переносом відбувається таким чином:



Послідовність дій при цьому така:

- крайній правий біт засилається в С;
- інші біти зсуваються управо;
- у крайній лівий розряд засилається первісний вміст регістра переносу С.

Зсуви уліво за технологією ідентичні, але робляться в зворотному напрямку. Їхні діаграми й алгоритми пропонується побудувати самостійно.

У системі команд МП КР580 передбачені такі команди зсуву:

- циклічний зсув уліво;
- циклічний зсув управо;
- зсув уліво через перенос;
- зсув управо через перенос.

Команди зсуву виконуються в акумуляторі над восьмирозрядними операндами. Результат заноситься в акумулятор.

3.2.1 Команди циклічного зсуву.

Команда циклічного зсуву уліво переміщує кожен біт у межах байта на один розряд уліво. При цьому вміст старшого розряду записується в молодший розряд і біт переносу. Команда циклічного зсуву управо переміщує кожен біт у межах байта на один розряд управо. При цьому вміст молодшого розряду записується в старший розряд і біт переносу.

- RLC - команда циклічного зсуву вліво;
- RRC - команда циклічного зсуву управо.

Як приклад, рекомендується виконати такі завдання:

а) записати в пам'ять коди програми, реалізуючої циклічний зсув байта на 4 розряди.

Таблиця 3.1

800	RLC	07	Циклічний зсув уліво на один розряд
801	RLC	07	----------//----------
802	RLC	07	----------//----------
803	RLC	07	----------//----------

Виконати програму, попередньо задаючи вихідні значення відповідно до табл. 3.2, командою <СТ> 800 <–> 804 <ВП>.

Таблиця 3.2

A вих.	00	0F	F0	81	A5	67
A рез.	00	F0	0F	18	5A	76

Перевірити отримані результати.

б) записати в пам'ять коди програми, реалізуючої операцію об'єднання старших тетрад двох байтів, що містяться в регістрах B і C, в один, використовуючи команду RRC.

Таблиця 3.3

800	MOV A, C	79	Пересилання A ← C
801	RRC	0F	Переміщення ст. тетради 1 байта
802	RRC	0F	на місце молодшої
803	RRC	0F	Переміщення ст. тетради 1 байта
804	RRC	0F	на місце молодшої
805	ANI FH	E6 0F	Вибір ст. тетради 1 байта
807	MOV C, A	4F	Пересилання C ← A
808	MOV A, B	78	Пересилання A ← B
809	ANI F0H	E6 F0	Виділення ст. тетради 2 байта
80B	ORA C	B1	Об`єднання двох байтів в один

Виконати програму, задаючи попередньо вихідні значення відповідно до табл. 3.4 командою <СТ> 800 <–> 80С <ВП>.

Таблиця 3.4

С	72	F0	51	19
В	9F	0F	A3	86
А рез.	97	0F	A5	81

Перевірити отримані результати.

3.2.2 Команди зсуву через перенос.

Команда зсуву уліво через перенос переміщує вміст кожного біта в межах байта на один розряд. При цьому вміст біта переносу записується в молодший розряд, а вміст старшого біта записується в біт переносу. Використовуючи цю команду, можна реалізувати операцію множення на числа кратні 2. Команда зсуву уліво через перенос переміщує вміст кожного розряду в межах байта управо на один розряд. При цьому в старший розряд байта записується значення байта переносу, а в нього заноситься вміст молодшого розряду байта. Використовуючи дану команду, можна реалізувати команду ділення на числа, кратні 2.

- RAL - команда зсуву уліво через перенос;
- RAR - команда зсуву управо через перенос.

Як приклад, рекомендується виконати такі завдання:

а) записати в пам'ять коди програми, що реалізує циклічний зсув уліво на один розряд вмісту пари регістрів HL.

Виконати програму, попередньо задаючи вихідні значення відповідно до табл. 3.6, командою <СТ> 800 <–> 80В <ВП>.

Перевірити отримані результати.

Таблиця 3.5

800	ORA A	B7	Скидання переносу в 0
801	MOV A, L	7D	A ← L
802	RAL	17	Зсув уліво на один розряд через перенос, в мол. розряд L - 0
803	MOV L, A	6F	L ← A
804	MOV A, H	7C	A ← H
805	RAL	17	Зсув уліво на один розряд через перенос
806	MOV H, A	67	H ← A
807	MOVA, L	7D	A ← L
808	ACI 0	CE 00	
80A	MOV L, A	6F	L ← A

Таблиця 3.6

HL вих.	FFFC	8002	3578	FFFF	0000	1111
HL рез.	FFF9	0005	6AF0	FFFF	0000	2222

б) записати в пам'ять коди програми, що реалізує операцію множення на 4 вмісту регістра C: B = C * 4

Таблиця 3.7

800	MOV A, C	79	A ← C
801	ORA A	B7	Скидання біта переносу
802	RAL	17	Множення на 2
803	RAL	17	Множення на 2
804	MOV B, A	47	Результат в B

Примітка: Вихідне значення не повинне перевищувати десяткове число 63. Виконати програму, попередньо задаючи вихідні значення відповідно до табл. 3.8, командою <СТ> 800 <–> 805 <ВП>.

Таблиця 3.8

C	00	02	10	2F	33	3A
B	00	08	40	BC	CC	E8

Перевірити отримані результати.

в) записати в пам'ять коди програми ділення на вісім вмісту регістра H. Цілу частину результату помістити в D, залишок в E.

Таблиця 3.9

800	MOV A, H	7C	A ← H
801	ORA A	B7	Скидання переносу в 0
802	RAR	1F	A = A/2
803	ORA A	B7	Перенос дорівнює 0
804	RAR	1F	A = A/2
805	ORA A	B7	Перенос дорівнює 0
806	RAR	1F	A = A/2
807	MOV D, A	57	D = H/8 - ціла частина
808	MOV A, H	7C	A ← H
809	ANI 07H	E6 07	Виділення залишку результату
80B	MOV E, A	5F	Залишок в E

Виконати програму, попередньо задаючи вихідні значення відповідно до табл. 3.9, командою <СТ> 800 <–> 80C <ВП>.

Таблиця 3.10

H	05	08	10	35	79	FF
D	00	01	02	06	OF	1F
E	05	00	00	05	01	07

Перевірити отримані результати.

3.3 Порядок виконання роботи

3.3.1 Написати і виконати програму об'єднання молодших тетрад двох байтів в один. Молодшу тетраду 2 байта помістити в старшу тетраду підсумкового байта. Використовувати: E - 1 байт, D - 2 байт, A - результат. Вихідні значення поєднуваних байт узяти довільно.

3.3.2 Написати і виконати програму, що реалізує операцію логічного множення трьох бітів (5,6,7) одного байта і трьох бітів (2,3,4) другого байта, попередньо перемістивши множені біти в молодші розряди й понуливши інші позиції. Прийняти позначення: L - 1 байт, H - 2 байт і C - результат. Вихідні значення прийняти довільно.

3.3.3 Написати і виконати програму циклічного зсуву на три розряди вмісту пари регістрів DE. Вихідні дані взяти довільно.

3.3.4 Написати і виконати програму множення вмісту пари регістрів HL на 4: HL = HL * 4. Вихідні дані взяти довільно.

3.3.5 Написати і виконати програму ділення вмісту пари регістрів BC на 8. BC = BC/8. Вихідні дані взяти довільно.

3.4 Зміст звіту

Звіт повинен містити:
- назву та мету роботи;
- написані програми відповідно до завдань 8.3;
- результати;
- висновки.

3.5 Контрольні запитання

3.5.1 Характеристика зсуву управо та уліво.
3.5.2 Особливості арифметичних зсувів управо та уліво.

3.5.3 Алгоритми циклічних зсувів.
3.5.4 Циклічні зсуви з переносом, їхні особливості.
3.5.5 Операції і команди зсуву МП КР580.
3.5.6 Практичне використання операцій зсуву, їхня корисність.

4 ЛАБОРАТОРНА РОБОТА № 9 ОПЕРАЦІЇ І КОМАНДИ ПОРІВНЯННЯ

4.1 Мета роботи

Вивчення операцій і команд порівняння мікропроцесора КР580.

4.2 Загальні відомості

У практиці дуже часті ситуації, коли виникає необхідність порівняння двійкових чисел. Найбільш типові вони при діагностиці правильності роботи МП (МПС) і прийнятті рішень у процесі обробки даних.

Операція порівняння складається в зіставленні двох будь-яких елементів даних.

Факт рівності двох чисел можна виявити за допомогою операцій " АБО, що виключає" і "Віднімання". Однак недоліком у даному випадку є часткова, або повна утрата вихідних даних при виконанні зазначених операцій. Зазначених недоліків позбавлена операція ПОРІВНЯННЯ. Вона заснована на відніманні вмісту регістра чи комірки пам'яті з вмісту акумулятора без зміни вихідних даних, тобто вмісту порівнюваних операційних елементів (комірка пам'яті, регістр, акумулятор) після виконання зазначеної операції.

У системі команд МП КР580 передбачено три типи команд порівняння:

- порівняння вмісту акумулятора і регістра;
- порівняння вмісту акумулятора і комірки пам'яті;
- порівняння вмісту акумулятора з безпосереднім операндом.

Команди порівняння виконуються за допомогою внутрішнього віднімання з вмісту акумулятора відповідно, вмісту регістра, комірки пам'яті чи безпосереднього операнда. Вміст акумулятора при цьому не

змінюється . Результатом порівняння є установка бітів ознак . Можливі варіанти установки ознак приведені в табл. 4.1.

Таблиця 4.1

Результат порівняння	Ознаки	
	Нуль	Перенос
Дорівнює	1	0
Більше	0	0
Менше	0	1

Крім перерахованих ознак установлюються так само біти парності і знака за результатами внутрішнього віднімання. Біт парності дорівнює 1 , у випадку коли кількість одиниць у результаті парна, і дорівнює 0 , якщо кількість одиниць непарна.

Біт знака встановлюється рівним значенню старшого розряду результату. Це дозволяє поставити процес виконання програми в залежність від значення результату виконання попередньої операції. Для звертання до інформації про результати обчислень у складі МП існує набір тригерів , що складають програмно-доступний регістр (прапорів) F , що встановлюються в одиницю чи скидаються в нуль у залежності від результату зроблених обчислень .Ознаки формуються за результатами арифметичних чи логічних операцій. Кожен тригер зберігає один з бітів умов. Програмно перевіряються значення чотирьох бітів умов:

а) S - знака результату (1 - мінус, 0 - плюс);

б) Z - нульового результату (1 - нульовий, 0 - не нульовий);

в) C - переносу (1 - перенос є, 0 - переносу немає);

г) P - паритет, парність (1 - число одиниць парне, 0 - непарне).

П'ятою ознакою є додатковий перенос AC з молодшої тетради в старшу при обробці двійково-десяткових чисел, які використовується разом з командою DAA.

Розташування ознак у розрядах РО таке:

F:	S	Z	0	AC	0	P	1	C
	7	6	5	4	3	2	1	0

Перевірка ознак результату здійснюється в двійковий формі представлення числа вмісту регістра F.

4.2.1 Команди порівняння з вмістом регістра такі:

CMP A (BF), CMP B (B8), CMP C (B9), CMP D (BA), CMP E (BB), CMP H (BC), CMP L (BD).

Як приклад, рекомендується виконати такі завдання:

- записати в пам'ять коди програми порівняння вмісту регістрів C і B;

Таблиця 4.2

800	MOV A, C	79	Пересилання A ← C
801	CMP B	B8	Порівняння A з регістром B

- виконати програму. попередньо задаючи вихідні значення відповідно до табл. 4.3, командою <СТ> 800 <–> 802 <ВП>;

Таблиця 4.3

	C	01	01	FF	43	55	3	20	AE	FF
	B	01	09	00	FF	55	55	15	B8	FF
Результат	F	56	83	96	07	56	83	02	97	56
	Нуль	1	0	0	0	1	0	0	0	1
	Перенос	0	1	0	1	0	0	0	1	0
	Знак	0	1	1	1	0	0	0	1	0
	Парність	1	0	1	1	1	1	0	1	1

- для перевірки зазначених у таблиці ознак вміст F переводиться з шістнадцяткової у двійкову форму представлення і фіксується вміст потрібних бітів.

4.2.2 Команда порівняння з вмістом комірки пам'яті:

CMP M (код BE) - порівняння вмісту регістра A і комірки пам'яті, адреса якої задана у парі HL.

Як приклад, рекомендується виконати такі завдання:

- записати в пам'ять коди програми порівняння вмісту регістра A і комірки пам'яті:

Таблиця 4.4

800	LXI H, 900H	21 00 09	Завантаження в HL. адреси 900
803	CMP M	BE	Порівняння A і M

- виконати програму, попередньо задаючи вихідні значення відповідно до табл. 4.5, командою <СТ> 800 <–> 804 <ВП>.

Таблиця 4.5

	A	21			BA			E9		
	M	00	21	39	19	FF	BA	E9	10	F5
Результат	F	16	56	87	92	87	56	56	92	93
	Нуль	0	1	0	0	0	1	1	0	0
	Перенос	0	0	1	0	1	0	0	0	1
	Знак	0	0	1	0	1	0	0	0	1
	Парність	1	1	1	0	1	1	1	0	0

- перевірити отримані результати.

4.2.3 Команда порівняння з безпосереднім операндом:

- CPI B (код FE) - порівняння вмісту регістра A з числом, заданим у другому байті команди, де B - байт.

Як приклад, рекомендується виконати такі завдання:

- записати в пам'ять коди команди порівняння з безпосереднім операндом;

Таблиця 4.6

800	CPI 7FH	FE 7FH	порівняння 7FH з A

- виконати програму, попередньо задаючи вихідні значення відповідно до табл.4.7, командою <СТ> 800 <–> 802 <ВП>;

Таблиця 4.7

	A	00	80	7F	B3	25	F7
Результат	F	87	02	56	02	87	06
	Нуль	0	0	1	0	0	0
	Перенос	1	0	0	0	1	0
	Знак	1	0	0	0	1	0
	Парність	1	0	1	0	1	1

- перевірити отримані результати.

4.3 Порядок виконання роботи

4.3.1 Скласти і виконати програму порівняння вмісту регістрів H і L. Скласти і заповнити таблицю вихідних даних і результатів. Вихідні дані взяти довільно. Вказати стан основних ознак.

4.3.2 Скласти і виконати програму порівняння вмісту регістра B і комірки пам'яті з адресою 906H. Заповнити таблицю вихідних даних і результатів з урахуванням ознак. Вихідні дані узяти довільно.

4.3.3 Скласти і виконати програму порівняння вмісту регістра H і безпосереднього операнда 5AH. Заповнити таблицю вихідних даних і результатів з урахуванням стану ознак. Вихідні значення регістра взяти довільно.

4.4 Зміст звіту

Звіт повинен містити:
- назву та мету роботи;

- написані програми відповідно до завдань 4.3;
- висновки.

4.5 Контрольні запитання

4.5.1 Особливості операції порівняння, її практична цінність.
4.5.2 Еквівалентні операції порівняння та їх недоліки.
4.5.3 Результати порівняння та їх визначення.
4.5.4 Перелічити ознаки з можливими значеннями бітів і короткою характеристикою.
4.5.5 Структура регістра ознак, її практичне використання.

5 ЛАБОРАТОРНА РОБОТА № 10 ОПЕРАЦІЇ, КОМАНДИ БЕЗУМОВНОГО ТА УМОВНОГО ПЕРЕХОДІВ

5.1 Мета роботи

Вивчення способів і прийомів реалізації переходів, вибору альтернатив.

5.2 Загальні відомості

Широкі можливості мікропроцесора багато в чому визначаються його здатністю приймати рішення. Команди переходу і виклику підпрограм є складовою частиною процесу прийняття рішень. У програмах керування завжди існує безліч галузей і хід виконання їх продовжується в тому чи іншому напрямку в залежності від значень параметрів контрольованого процесу. Команди переходу і виклику підпрограм дозволяють змінювати послідовність виконання команд програми за рахунок зміни вмісту лічильника команд. Існує два способи зміни зазначеної послідовності. Перший з них називають безумовним. Відповідно до нього послідовність виконання програми піддається зміні всякий раз, коли реалізується визначена команда. Відповідно до іншого способу послідовність виконання програми визначається деякими умовами і змінюється тільки тоді, коли значення зазначеної умови збігається з заданим.

Принцип організації такий. У результаті виконання команд порівняння, арифметичної і логічної обробки даних змінюються значення розрядів нульового, від'ємного результатів і переносу регістра стану. Команди переходу і виклику підпрограм перевіряють значення розрядів регістра стану і визначають наступний хід виконання програми в залежності від результатів перевірки. Команди переходу (альтернативні команди), називають також командами розгалужування, вони дозволяють організувати в підпрограмах цикли і розгалужувння. Вони є засобом зміни значення вмісту лічильника команд і, отже, зміни нормальної послідовності виконання програми.

У системі команд МП КР580 передбачені команди зміни послідовності виконання команд для організації циклів, обробки умов, передачі керування і т.п. Існує два типи команд переходу: безумовний і умовний.

При виконанні команди безумовного переходу здійснюється передача керування за адресою, заданою в другому і третьому байтах команди, або за адресою, заданою в регістровій парі. Команди умовного переходу виконуються в тому випадку, якщо встановлений чи скинутий відповідний біт ознаки, у противному випадку команда ігнорується і виконується наступна за нею команда. Команди умовного переходу існують для таких бітів ознак:

- біта нуля;
- біта переносу;
- біта знака;
- біта парності.

Для кожного біта ознаки передбачені дві команди переходу: перехід по встановленій ознаці (логічна одиниця) і перехід по скинутому біті ознаки (логічний нуль).

Відповідність виконання команди і стани регістру ознак наиведені в табл. 5.1.

Таблиця 5.1 - Відповідність команд бітам ознак

Ознака стану	Нуль Z		Перенос C		Парність P		Знак S	
Команда	1	0	1	0	1	0	1	0
JZ	+	-	-	-	-	-	-	-
JNZ	-	+	-	-	-	-	-	-
JC	-	-	+	-	-	-	-	-
JNC	-	-	-	+	-	-	-	-
JPE	-	-	-	-	+	-	-	-
JPO	-	-	-	-	-	+	-	-
IM	-	-	-	-	-	-	+	-
IP	-	-	-	-	-	-	-	+

Знак (+) відповідає “так”, знак (-) відповідає “ні”.

5.2.1 Команди безумовного переходу такі:

- JMP add r - безумовний перехід за адресою, зазначеною в другому та третьому байтах команди;
- PCHL - безумовний перехід за адресою, заданою в HL.

Як приклад, рекомендується виконати такі завдання:

- записати в пам'ять з адреси 800H коди програми, що здійснює перехід за адресою, записаною в масиві. Адреса є елементом масиву і вибирається відповідно до заданого індексу масиву. Масив розміщується в пам'яті з адреси 810H і містить п'ять елементів. Індекс задається регістром C.

Виконати програму поетапно, попередньо задаючи вихідні значення та адресу точки зупину (ТЗ) - друга адреса в команді монітора <Старт програми> відповідно до табл. 5.4, <СТ> 800 <–> ТЗ <ВП>.

Порівняти вміст комірки пам'яті з вказаним в програмі.

Таблиця 5.3

800	LXI D, 810H	11 10 08	Завантажити DE начальною адресою таблиці
803	MOV L, C	69	Пересилання L <- C - індекс
804	MVI H, 0	26 00	Завантажити HL індексом елемента масиву
806	DAD H	29	HL = HL*2 зміщування таблиці
807	DAD D	19	HL = база + зміщування
808	MOV A, M	7E	Пересилання A ← M молодшого байта адреси переходу
809	INX H	23	Зміна адреси на 1
80A	MOV H, M	66	Пересилання A ← M старшого байта адреси переходу
80B	MOV L, A	6F	Пересилання L ← A молодшого байта адреси переходу
80C	PC HL	E9	Перехід, завантажити PC = HL
80D	DW 900H	00 09	Масив адрес переходів
80F	DW 902H	02 09	- “ -
811	DW 904H	04 09	- “ -
813	DW 906H	06 09	- “ -
815	DW 908H	08 09	- “ -

Таблиця 5.4

C	00	01	02	03	04
T31 - 806 H Зміщення (HL)					
T32 - 807H Адреса елемента масиву (HL)					
T33 – 80BH Адреса переходу (HL)					
T34	900	902	904	906	908

5.2.2 Команди переходу за ознакою Z – нуль такі:
- JZ addr - перехід, якщо Z = 1;
- JNZ addr - перехід, якщо Z = 0.

Як приклад, рекомендується виконати такі завдання:

а) записати в пам'ять, починаючи з адреси 800H, програму заповнення 10H комірок пам'яті нулями.

Таблиця 5.5

800	MVI C, 10H	OE 10	Завантаження C=10H, довжина масиву
802	LXI H, 900 H	21 00 09	Завантаження HL=900H початковою адресою масиву
805	MVI M, 0H	36 00	Завантаження M=0
807	INX H	23	HL=HL+1, наступна адреса
808	DCR C	OD	C=C-1, довжина масиву
809	JNZ 805 H	C2 05 08	Перехід, якщо Z=0

Виконати програму командою <СТ> 800 <–> 80C <ВП>.
Перевірити результати виконання відповідно до табл. 5.6.

Таблиця 5.6

C	H	M (900H-90FH)
00	910	00

б) записати в пам'ять з адреси 800H програму заміни даних у масиві пам'яті (900H - 90FH), відмінних від 00H на 00H.

Таблиця 5.7

800	MVI C, 10H	0E 10	Завантажити C=10H, довжина масиву
1	2	3	4
802	LXI H, 900H	21 00 09	Завантажити HL = 900H, поч.адреса масиву
805	MOV A, M	7E	Читання A ← M
806	CPI O	FE 00	A=0 ?
808	JZ 80DH	CA 0D 08	Перехід, якщо ТАК

Продовження табл. 5.7

1	2	3	4
80B	MVI M, 0	36 00	якщо HI, завантажити M=0
80D	INX H	23	HL=HL + 1, наступна адреса
80E	DCR C	0D	C = C - 1, довжина масиву
80F	JNZ 805H	C2 05 08	Продовжувати, якщо довжина масиву 0

Виконати програму командою <СТ> 800 <–> 812 <ВП>.
Перевірити результати відповідно до табл. 5.6.

5.2.3 Команди переходу за ознакою С – перенос такі:
- JC addr - перехід, якщо С = 1;
- JNZ addr - перехід, якщо С = 0.

Як приклад, рекомендується виконати такі завдання:

а) записати в пам'ять з адреси 800H програму підрахунку нулів у байті. Вихідне значення задане в регістрі С.

Таблиця 5.8

800	MOV A, C	79	A ← C
801	MVI B, 8	06 08	Завантажити B=8, кількість розрядів у байті
803	MVI E, 0	1E 00	Завантажити E=0, вихідне значення кількості нулів
805	RAR	1F	Зсув управо і мол. біт у біт переносу
806	JC 80AH	DA 0A 08	Якщо C=1 обійти инкремент рахунку нулів
809	INR E	1C	E=E+1, інкремент рахунку нулів
80A	DCR B	05	B=B-1, наступній розряд
80B	JNZ 805	C2 05 08	Перехід на перевірку наступного розряду байта

Виконати програму, попередньо задаючи вихідні значення відповідно до табл. 5.9, командою <СТ> 800 <–> 80E <ВП>.

Таблиця 5.9

C	00	DB	FF	03	9A	55
E	00	06	08	02	04	04

Перевірити результати.

б) записати в пам'ять з адреси 800H програму порівняння двох шістнадцятирозрядних величин з врахуванням більше або рівно: HL > DE, C = 0, інакше C = 1.

Таблиця 5.10

800	MVI C, 0	0E 00	Завантажити C = 0, якщо HL > DE
802	MOV A, L	7D	A ← L
803	SUB E	93	Порівняння молодших байтів, якщо L < E, перенос встановлюється в 1
804	MOV A, H	7C	A ← H
805	SBB D	9A	Порівняння старших байтів з урахуванням переносу
806	JNC H	D2 0B 08	HL > DE, перенос 0, C = 0
809	MVI C, 1	0E 01	HL < DE, перенос 1, C = 1

Виконати програму, попередньо задаючи вихідні значення відповідно до табл. 5.11, командою <СТ> 800 <–> 80B <ВП>.

Таблиця 10.11

HL	0000	FFFF	8002	5A7F	3A55	0000
DE	0001	FFFF	7FFF	6080	C201	FFFF
C	01	00	00	01	01	01

Перевірити результати.

5.2.4 Команди переходу за ознакою P – парність

- JPE addr - перехід, якщо P = 1;

- JPO addr - перехід, якщо P = 0.

Як приклад, рекомендується виконати такі завдання:

- записати в пам'ять з адреси 800H програму доповнення байта до парності в стартовому розряді. Вихідне число в регістрі C;

Таблиця 5.12

800	MOV A, C	79	Пересилання A ← C, вихідний байт
801	ANI 7FH	E6 7F	Обнулення старшого розряду
803	ORA A	B7	A OR A, встановлення біта парності
804	JPO 809H	E2 09 08	Перехід, якщо початковий байт парний
807	ORI 80H	F6 80	Доповнити до парності
809	MOV C, A	4F	C ← A, результат

- виконати програму, задаючи вихідні значення відповідно до табл. 5.13, командою <СТ> 800 <–> 80A <ВП>;

Таблиця 5.13

C вих.	00	FF	B6	80	CD	75
C рез.	00	FF	36	00	4D	F5

- перевірити результат.

5.2.5 Команди переходу по ознаці S – знак такі:

- JM addr - перехід, якщо S=1 (число < 0);
- JP addr - перехід, якщо S = 0 (число > 0).

Як приклад, рекомендується виконати такі завдання:

- записати в пам'ять з адреси 800H, коди програми виділення з масиву 100H - 10FH цифрових символів з кодами 30H - 39H і перезапису їх в інший масив з початковою адресою 900H;

Таблиця 5.14

800	LXI H, 100H	21 00 01	Завантажити HL поч. адресою вихідного масиву
803	MVI B, 0FH	06 0F	Завантажити B - 0FH довжиною масиву
805	LXI D, 900H	11 00 09	Завантажити DE поч. адресою кінцевого масиву
808	MOV A, M	7E	Читання A ← M
809	SBI 30H	DE 30	Перевірка 30H < A < 39H A=A-30H, якщо A < 00H, S=1
80B	JMP 805H	FA 05 08	Перехід до наступного елементу
80B	SBI 0AH	DE 0A	A > 30H, A=A-0A, якщо A > 0, S=0
80F	JP 805H	F2 05 08	Перехід до наступного елементу
812	MOV A, M	7E	30H < A < 39H
813	STAX	12	Передати в масив виведення
814	INX D	13	Адреса наступного елементу кінцевого масиву
815	INX H	23	Адреса наступного вихідного масиву
816	DCR B	05	Довжина масиву дорівнює одному
817	JNZ 808H	C2 08 08	Якщо ні, продовжувати

- виконати програму, перевірити правильність її роботи. Для цього переглянути кінцевий масив (900H - 90FH).

Значення елементів масиву (ЕМ) повинні відповідати умові 30H < ЕМ < 39H.

5.3 Порядок виконання роботи

5.3.1 Вивчити команди й особливості їхнього використання при реалізації безумовних переходів, а також з використанням ознак Z, C, P, S;

5.3.2 Вивчити і виконати завдання прикладів.

5.3.3 Реалізувати програму для нескінченного циклу підсумовування вмісту масиву пам'яті з адресами 0800H - 08FFH;

5.3.4 Реалізувати програмний перехід за адресою, що обчислюється по формулі:

АП = БА + І * 16,

де АП - адреса переходу;
БА - базова адреса (БА = 920Н);
І - індекс, що задається в регістрі Е.

Встановлюючи точки зупину у виконуваній програмі, заповнити табл. 5.15

Таблиця 5.15

Е	0	2	5	7	10
СМ=N*16					
АП					

5.3.5 Реалізувати програму заповнення масиву 900Н-9FFH у такий спосіб: у парні комірки (900Н, 902Н, 904Н і т.д.) записати 55Н, а в непарні (901Н, 903Н, 905Н і т.д.) записати ААН.

5.3.6 Реалізувати програму підрахунку контрольної суми масиву 900Н - 9FFH по модулю 256 (без урахування переповнення байта).

5.3.7 Реалізувати програму підрахунку кількості одиниць і нулів у байтах масиву пам'яті, початкові і кінцеві адреси якого задаються відповідно в парах HL і DE. Заповнити таблицю 5.16.

Таблиця 5.16

HL	0800	0800	0C00
DE	08FF	0AFF	FFFF
Число 1			
Число 0			

5.3.8 Реалізувати програму доповнення байта до парності і заповнити табл. 5.17

Таблиця 5.17

С вих.	00	FF	B6	80	CD	75
С рез.						

5.3.9 Реалізувати програму підрахунку в масиві кількості парних і непарних байтів. Початкова і кінцева адреси задаються в парах регістрів HL і DE, відповідно. Заповнити табл. 5.18.

Таблиця 5.18

HL	0800	0800	0C00
DE	08FF	0AFF	FFFF
Парних			
Непарних			

5.3.10 Реалізувати програму, формуючи два масиви з одного за такими умовами, табл. 5.19.

Таблиця 5.19

Вихідний масив	Початкова адреса	Кінцева адреса	Елемент масиву
	0	7FH	
1 масив	900H		0H < EM < 7FH
2 масив	980H		80H < EM < FFH

Підрахувати кількість елементів у кожнім масиві.

Перевірити правильність роботи програми переглядом відповідних комірок обох масивів.

5.4 Зміст звіту

Звіт повинен містити:
- назву та мету роботи;
- написані програми відповідно до завдань 5.3;
- результати та висновки.

5.5 Контрольні запитанння

5.5.1 Призначення, види й особливості операцій розгалужування.

5.5.2 Прийняття рішень і альтернативи в програмуванні.

5.5.3 Принцип організації чи циклу розгалужування в програмі.

5.5.4 Особливості використання умов при переходах.

5.5.5 Техніка виконання команд переходів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Микропроцессоры. В 3-х книгах. Учебник для ВТУЗов/ П.В. Нестеров, В.Ф. Шаньгин, В.Д. Горбунов и др.; под ред. Л.Н. Преснухина. – М.: Высш. шк., 1986.

2. Стрыгин Б.В., Щарев Л.С. Основы вычислительной, микропроцессорной техники и программирования. – М.: Высш. шк., 1989. – 479 с.

3. Каган Б.М., Сташин В.В. Микропроцессоры в цифровых системах. - М.: Энергия, 1979.

4. Грибанов В.П. и др. Операционные системы: Учеб. пособие М.: Финансы и статистика, 1990.-239с.

5. Справочник по микропроцессорным устройствам. /А.А. Молчанов, В.И. Корнейчук, В.П. Тарасенко, Д.А. Россошинский. - К.: Техника, 1987. – 288с.

6. Погорелый С.Д., Слободянюк Т.Ф. Программное обеспечение микропроцессорных систем: Справочник. – 2-е изд. К.: Техника, 1989. – 301с.

7. Токхайм Р. Микропроцессоры: курс и упражнения, /пер. с англ., под ред. В.Н. Грасевича. - М.: Энергоатомиздат, 1987. – 336с.

8. Гилмор Ч. Введение в микропроцессорную технику. - М.: Энергоатомиздат, 1983.-464с.

9. Майоров С.А. и др. Введение в микроЭВМ. - Л.: Машиностроение, 1988.- 304с.

10. Майоров В.Г., Гаврилов А.И. Практический курс программирования микропроцессорных систем. - М.: Машиностроение, 1989. – 272 с.