;] ASTRA-256

БІНАРНИЙ КОМП’ЮТЕР ПРОГРАМОВАНИЙ НА АСЕМБЛЕРІ

ЗМІСТ

ПРИЗНАЧЕННЯ ІНДИКАТОРІВ

Індикатор	Призначення
🟢 **AD7** – 🟢 **AD0**	Регістр адреси (аналог стану шины адреси). Відображає поточну адресу елемента пам'яті, вміст якого можна змінити вручну. В процесі виконання програми відображає вміст лічильника команд.
🟢 **DI7** – 🟢 **DI0**	в режимі "DI view" відображають стан регістра введення даних DI. (В режимі "DI view" засвічується індикатор 🟢 **DI**). Якщо індикатор ● **DI** не горить, то 🟢 **DI0** – 🟢 **DI7** відображають стан елемента пам'яті, що адресується 🟢 **AD0** – 🟢 **AD7**.
🟢 **DO7** – 🟢 **DO0**	відображає стан регістра DO. Використовується для виведення результату обчислень. У режимі відладки "Debug" 🟢 **DO0** – 🟢 **DO7** можуть використовуватись для відображення вмісту акумулятора. (В цьому випадку включений індикатор 🟡 **ACC**).
🟢 **RUN**	показує, що пристрій знаходиться в процесі виконання програми. Засвічується після натискання кнопки **RUN/STOP**.
🟡 **ACC**	сигнал про те, що 🟢 **DO0** – 🟢 **DO7** відображають стан акумулятора. Використовується тільки в режимі "Debug", після натискання на кнопку **DO/A/<--**.
🟡 **DI**	сигнал того, що на 🟢 **DI0** – 🟢 **DI7** відображається стан регістра DI. DI зазвичай використовується для введення даних.
🔴 **HALT**	засвічується червоним при виконанні команди HLT. Команда HALT тимчасово призупиняє роботу програми до натискання клавіші **ENTER**. (окільки машина при цьому знаходиться в стані виконання програми, індикатор 🟢 **RUN** світиться). Засвічується жовтим при виконанні команди INKBD.
🔵 **DEBUG**	світиться, якщо машина знаходиться в стані відладки "Debug".
🔵 **BIN**	світиться, якщо машина знаходиться в стані введення даних в бінарному форматі.

ПРИЗНАЧЕННЯ КНОПОК

Кнопка	Призначення
0 – **7**	Використовуються для введення адреси в регістр адреси, або даних в елемент пам'яті, адресований регістром адреси, або в регістр вхідних даних Di. Якщо встановлено режим "Бінарного введення" (при цьому світиться індикатор ● **BIN**), то ці кнопки змінюють стан регістрів ● **AD0** – ● **AD7** або ● **DI0** – ● **DI7**, що знаходяться над ними, в залежності від того, яку кнопку було натиснуто (**SLCT ADR** / **SAVE DATA** / **SAVE DI**). При натисканні однієї з цих кнопок інвертується значення біта адреси ● **AD0** – ● **AD7**, що знаходиться над кнопкою, або біта даних ● **DI0** – ● **DI7**. Ці кнопки є аналогом тумблерів, які використовували "справжні програмісти" для введення в пам'ять даних і програм. Якщо натисканням клавіші **BIN/HEX** встановлено режим "16-річного введення" (при цьому індикатор ● **BIN** не світиться), то кнопки **0** – **7** виконують роль перших 8 кнопок для введення значення байта у 16-річному форматі. У багатьох сучасних комп'ютерах, включаючи серію PC, для запису чисел в двійковій системі прийнято порядок запису розрядів - зліва на право, від старших бітів до молодших. Тому в нашому комп’ютері порядок розміщення кнопок від **0** до **7** на клавіатурі нетрадиційний. Це зроблено для того, аби кожна з кнопок була розташована безпосередньо під тим бітом-індикатором, значення якого вона змінює.
0 – **F**	призначені для введення значень в 16-річному форматі. Якщо встановлено режим "16-ічне введення" (при цьому світиться індикатор ● **BIN**). Перше натискання однієї з цих кнопок встановлює старші чотири біта, друге натиснення встановлює значення молодших чотирьох бітів даних, що вводяться.
A – **Z**	Клавіатура. Слугує для введення команд на Assembler. Крім того, з цими клавішами працюють команди, які повертають код натиснутих клавіш (див. Команди введення / виведення).
BIN/HEX	Кнопка **BIN/HEX** "бінарне/16-річне введення". Якщо встановлено режим "Бінарне введення" (при цьому світиться індикатор ● **BIN**), стан регістрів ● **AD0** – ● **AD7** або ● **DI0** – ● **DI7** змінюється при натисканні кнопок **0** – **7**,

	розташованих під ними (в залежності від того, яку кнопку було перед цим натиснуто **SLCT ADR** / **SAVE DATA** / **SAVE DI**). Якщо встановлено режим "16-річне введення" (при цьому індикатор ● **BIN** не світиться), то введення значення проводиться в 16-річному форматі кнопками **0** – **F**.
DO/A/<--	Клавіша "Backspace" витирає попередній введений символ в режимі редагування команд Асемблера (**ASM** --> **SAVE DATA**). В інших випадках натискання цієї клавіші переводить індикатори ● **DO0** – ● **DO7** в режим відображення вмісту акумулятора АС. При цьому засвічується індикатор ● **ACC**. Повторне натискання на клавішу повертає ● **DO0** – ● **DO7** в режим відображення регістра DO і гасить ● **ACC**.
DI VIEW	режим відображення на індикаторах ● **DI0** – ● **DI7** вмісту регістра вхідних даних DI. (При цьому засвічується індикатор ● **DI**). Якщо цей режим вимкнено, то на ● **DI0** – ● **DI7** завжди відображається вміст елемента пам'яті, що адресується (шиною адреси) ● **AD0** – ● **AD7**.
SLCT ADR	при натисканні на цю кнопку комп'ютер переходить в режим введення адреси (задіяно індикатори ● **AD0** – ● **AD7**). Адресу можна: • Ввести з **0** – **7** в режимі "Bin". • Ввести з **0** – **F** в режимі "Hex". • Збільшити на 1, натиснувши на клавішу **→**. • Зменшити на 1, натиснувши на клавішу **←**. При цьому, якщо вимкнено режим "Di View" (індикатор ● **DI** не світиться), то на індикаторах ● **DI0** – ● **DI7** відображається стан елемента пам'яті, що адресується ● **AD0** – ● **AD7**. Таким чином, ви завжди маєте можливість переглянути вміст елемента пам'яті із заданою адресою.
ENTER	завершення введення адреси або даних, (в залежності від раніше натиснутої **SLCT ADR** / **SAVE DATA** / **SAVE DI**), із записом значення в обраний регістр. Наприклад, якщо раніше було натиснуто клавішу **SLCT ADR**, то робиться запис в регістр адреси.
SAVE DATA	В режимі "MEM": при натисканні на цю кнопку комп'ютер переходить в режим введення даних в елемент пам'яті, що адресується (задіяно індикатори ● **DI0** – ● **DI7**). Адресу можна: • Ввести з **0** – **7** в режимі "Bin". • Ввести з **0** – **F** в режимі "Hex". В цьому випадку при натисканні на клавішу **ENTER** значення, що

	відображається на ● **DI0** – ● **DI7** записується в елемент пам'яті, що адресується ● **AD0** – ● **AD7**. Увага! Після закінчення роботи програми **SAVE DATA** значення регістра адреси автоматично збільшується на 1, для зручності введення наступної команди. Вихід з режиму **SAVE DATA** залежить від установки параметра "ВИХІД ІЗ SAVE DATA ПО ENTER" в розділі "Налаштування" і може здійснюватись або натисканням клавіші **ENTER** або повторним натисканням **SAVE DATA**. В режимі "ASM": при натисканні на цю кнопку комп’ютер переходить в режим введення команд на мові Асемблер. Перехід до редагування наступної команди здійснюється після натискання клавіші **ENTER**. При цьому значення регістра адреси автоматично збільшується на кількість байт введеної команди. Вихід з режиму введення команд здійснюється повторним натискання клавіші **SAVE DATA**.
SAVE DI	запис даних в регістр DI. Відрізняється від **SAVE DATA** тим, що при натисканні на кнопку **ENTER** дані записуються в регістр вхідних даних DI. Після закінчення роботи програми **SAVE DI** значення регістра адреси не змінюється.
SAVE PROG	запис програми (по суті - дампа пам'яті) на віртуальний носій. При натисканні на цю клавішу йде запит на введення імені програми. Запис програми в файл із вказаним ім'ям проводиться при натисканні на **ENTER**.
LOAD PROG	читання програми (по суті - дампа пам'яті) з віртуального носія в пам'ять. При натисканні на цю клавішу йде запит на введення імені програми. Читання програми з файлу із вказаним ім'ям проводиться при натисканні на **ENTER**.
RUN/STOP	запуск / зупинка програми. Запуск починаючи з поточної адреси AD. Якщо в момент запуску машина перебувала в режимі відладки "Debug", то машина виходить з цього режиму. При запуску засвічується індикатор ● **RUN**. Якщо програму було попередньо запущено, при натисканні цієї кнопки виконання програми припиняється.

DBG	переведення комп’ютера в режим "відладки" - покрокового виконання команд. При цьому засвічується індикатор ● **DEBUG**. В цьому режимі команди виконуються покроково. Функцію команди «крок вперед» виконує клавіша **␣**. Вихід з режиму здійснюється або натисканням клавіші **DBG**, або натисканням клавіші **RUN / STOP**.
MEM	відображення на екрані стану елементів пам'яті пристрою. Цей режим дозволяє спостерігати, як змінюються значення регістрів та елементів пам'яті в процесі виконання програми.
HELP VIEW	відображення на екрані довідника команд. При натисканні на клавіші **A** – **Z** виводиться список команд, що починаються з відповідної літери, з їх коротким описом. Вихід з режим здійснюється натисканням на будь-яку функціолнальну клавішу.
ASM	Перехід в режим відображення послідовності команд Assembler. При цьому в рядку послідовно відображається: • Адреса, за якою знаходиться команда; • Символьна назва команди; • Другий, третій і четвертий байти команди, якщо команда 2-х, 3-х або 4-х байтна відповідно. В цьому режимі, при натисканні на клавішу **SAVE DATA** з'являється можливість редагувати команду, записуючи її назву з клавіатури.
MENU	Вхід в меню застосунку. Вибір режимів: роботи, налаштувань або довідки.
EXIT	Вихід із застосунку. **Увага!** Перед виходом із застосунку збережіть свою роботу в файл за допомогою клавіші **SAVE PROG**. (Після підтвердження виходу **дані автоматично не зберігаються**.)

СТРУКТУРА ВІРТУАЛЬНОЇ МАШИНИ

Віртуальна машина є 8-бітним бінарним комп'ютером, який має архітектуру фон Неймана з набором із 75 команд.

В розпорядженні програміста є окремий 8-розрядний регістр - акумулятор AC, з вмістом якого можна проводити арифметичні та логічні операції, і оперативна пам'ять.

Об'єм оперативної пам'яті 256 байт.

У старших адресах оперативної пам'яті є також набір спеціальних 8-розрядних регістрів:

- покажчик стека SP;
- регістр прапорів FR;
- регістр введення даних DI;
- лічильник команд IP;
- регістр виведення даних DO.

РЕГІСТРИ

Акумулятор **AC**. Призначений для виконання арифметичних і логічних операцій над вмістом. Вміст цього регістра може бути виведено на індикатори ● **DO0** – ● **DO7**, якщо в режимі "Debug" було натиснуто кнопку **DO/A/<--**.

Регістри віртуальної машини, що відображені на пам'ять. І розміщені у верхній області пам'яті з адреси 251 по 255.

Регістр - покажчик стека SP (знаходиться в пам'яті за адресою 251 (11111011)). У цьому регістрі міститься поточна адреса вершини стека.

- При записі байта в стек, значення SP зменшується на 1 і байт записується в елемент пам'яті, що адресується SP.
- При вилученні байта зі стека операція проводиться у зворотному порядку. Байт зчитується з елементу пам'яті, що адресується SP, після чого сам SP збільшується на 1.

Регістр прапорів FR (знаходиться в пам'яті за адресою 252 (11111100)). Окремі біти FR мають окреме призначення. Призначення бітів регістра прапорів FR:

- **ZF** - прапор нуля. Якщо він = 1, то це значить, що виконання попередьої інструкції в акумуляторі дало відповідь = 0;
- **CF** - прапор перенесення. Він = 1, якщо попередня інструкція в акумуляторі дала біт перенесення (переповнення) або від'ємний результат. Інакше = 0;
- **TF** - прапор переповнення лічильника адреси (= 0, якщо працює в штатному режимі. = 1, якщо трапилося переповнення лічильника адреси).

Регістр введення даних DI (знаходиться в пам'яті за адресою 253 (11111101)). В DI зберігаються дані, які "справжні програмісти" вводили "з тумблерів". Вміст цього регістра відображається на індикаторах ● **DI0** – ● **DI7**.

Регістр лічильника команд IP (знаходиться в пам'яті за адресою 254 (11111110)). Вміст цього регістра можна прочитати або записати "з тумблерів". У регістрі знаходиться поточний адресу, з якого обрана команда для виконання. Після виконання поточної команди, вміст регістра автоматично змінюється (на кількість байт, яку займає команда), вказуючи на адресу наступної команди. Вміст цього регістра відображається на індикаторах ● **AD7** – ● **AD0**.

Регістр виведення даних DO (знаходиться в пам'яті за адресою 255 (11111111)). В цей регістр, як правило, виводиться результат, або точніше - певний окремий байт результата обчислень. Вміст цього регістра виводиться на індикатори ● **DO7** – ● **DO0**.

СИСТЕМА КОМАНД

Команди можуть бути 1-байтні, 2-х байтні або 3-х байті (також є три 4-х байтні команди). Тобто, при виконанні таких команд, лічильник команд IP збільшуватиметься на 1, 2, 3 або 4 відповідно, для вибору адреси наступної команди.

Команди, що керують процесом виконання програми
NOP | **00h** | 00000000b | 1 byte Немає операції (нічого не робить, "порожня" команда).
SPEED | **02h** | 00000010b | 2 bytes Встановити швидкість, з якою віртуальний процесор виконуватиме кожну інструкцію після запуску програми. 2-й байт - параметр гальмування швидкості виконання програми.
ADDRIP | **03h** | 00000011b | 2 bytes | TF Додати вміст зазначеного элемента пам'яті до значення лічильника команд IP. Залишити результат в IP. 2-й байт - адреса елемента пам'яті, вміст якого потрібно додати до IP. Якщо при виконанні команди відбувається переповнення регістра лічильника команд IP, то прапор TF = 1, інакше TF = 0.
HLT | **0Eh** | 00001110b | 1 byte Призупинити виконання інструкцій. При цьому засвічується індикатор ● **HALT** червоного кольору. Виконання програми продовжується при натисканні на кнопку **ENTER** або **RUN/STOP**. Зазвичай використовується для очікування введення значення в регістр DI з клавіатури.
STOP | **0Fh** | 00001111b | 1 byte Зупиняє виконання програми до натискання клавіші **RUN/STOP**.
Команди копіювання до або з акумулятора
MOVLA | **10h** | 00010000b | 2 bytes | ZF Записати байт в акумулятор AC.

2-й байт - значення, яке буде занесено в акумулятор.
Якщо в результаті операції в акумуляторі отримуємо 00h, тоді прапор ZF = 1, інакше ZF = 0.

MOVRA | **11h** | 00010001b | 2 bytes | ZF

Скопіювати в акумулятор вміст елемента пам'яті.
2-й байт - адреса елемента пам'яті, дані з якого будуть записані в акумулятор.
Якщо в результаті операції в акумуляторі отримуємо 00h, тоді прапор ZF = 1, інакше ZF = 0.

MOVAR | **12h** | 00010010b | 2 bytes | ZF

Скопіювати вміст акумулятора в зазначений елемент пам'яті.
2-й байт - адреса елемента пам'яті.
Якщо в результаті операції в акумуляторі отримуємо 00h, тоді прапор ZF = 1, інакше ZF = 0.

MOVIRA | **13h** | 00010011b | 2 bytes | ZF

Копіює в Акумулятор вміст елемента пам'яті, адреса якого знаходиться в іншому елементі пам'яті.
2-й байт - адреса елемента пам'яті, в якому знаходиться адреса іншого елемента пам'яті.
Якщо в результаті операції в Акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

MOVIAR | **14h** | 00010100b | 2 bytes

Копіює вміст Акумулятора в елемент пам'яті, адреса якого знаходиться в іншому елементі пам'яті.
2-й байт - адреса елемента пам'яті, в якому знаходиться адреса іншого елемента пам'яті.

MOVILR | **15h** | 00010101b | 3 bytes

Записує буквальне значення в елемент пам'яті, адреса якого знаходиться в іншому елементі пам'яті.
2-й байт - значення, що має бути занесено в елемент пам'яті.
3-й байт - адреса елемента пам'яті, в якому міститься адреса іншого елемента пам'яті, що в нього має бути скопійовано значення.

MOVAL | **16h** | 00010110b | 2 bytes | ZF

Записує вміст Акумулятора в другій байт цієї ж команди.
2-й байт - байт, в який буде записано вміст Акумулятора. Може мати будь-яке початкове значення. Воно зміниться в процесі виконання команди.
Якщо результат в Акумуляторі 00h, то прапор ZF = 1, інакше ZF = 0.

LOIRA | **17h** | 00010111b | 2 bytes | ZF

Записує в Акумулятор число з елемента пам'яті, адреса якого міститься в іншому елементі пам'яті. Після цього, в залежності від значення прапора CF, збільшує або зменшує на 1 значення адреси, що міститься в зазначеному елементі. Якщо CF = 0, адреса збільшується на 1, якщо CF = 1, адреса зменшується на 1.
2-й байт - адреса елементу пам'яті, в якому міститься адреса для виконання непрямої адресації. Після виконання операції, вміст цього елемента пам'яті збільшується або зменшується на 1.
Якщо в результаті операції в Акумуляторі отримуємо 00h, тоді прапор ZF = 1, інакше ZF = 0.

CLEARA | **E4h** | 11100100b | 2 bytes | ZF

Переносить вміст Акумулятора по вказаній адресі, після чього обнуляє Акумулятор.
2-й байт - адреса елементу пам'яті, до якого буде записано вміст Акумулятора.
Завжди встановлює ZF = 1.

Команди копіювання в пам'ять

MOVLR | **20h** | 00100000b | 3 bytes

Записати байт в пам'ять за вказаною адресою.
2-й байт - дані, які буде записано в пам'ять.
3-й байт - адреса елемента пам'яті, куди буде записано дані.

MOVRR | **21h** | 00100001b | 3 bytes

Скопіювати значення з одного елемента пам'яті в інший.
2-й байт - адреса елемента пам'яті, з якого буде прочитано дані.
3-й байт - адреса елемента пам'яті, в який буде записано дані.

MOVIRR | **22h** | 00100010b | 3 bytes

Копіює вміст однієї елемента пам'яті в інший, при непрямій адресації обох елементів.
2-й байт - адреса елемента пам'яті з адресою іншого елемента пам'яті, з якого копіюється значення.
3-й байт - адреса елемента пам'яті з адресою іншого елемента пам'яті, в який має бути записано значення.

CLEARR | **E5h** | 11100101b | 2 bytes

Записує нуль в адресований елемент пам'яті.

2-й байт - адреса елемента пам'яті, в який буде записано 00h.

Команди обміну

XCHGRA | **30h** | 00110000b | 2 bytes | ZF

Міняє місцями вміст акумулятора і вміст зазначеного елемента пам'яті.
2-й байт - адреса элемента пам'яті, з яким виконується обмін даними.
Якщо в результаті операції в акумуляторі отримуємо 00h, тоді прапор ZF = 1, інакше ZF = 0.

XCHGRR | **31h** | 00110001b | 3 bytes

Міняє місцями вміст пари адресованих елементів пам'яті.
2-й байт - адреса елемента пам'яті, з яким виконується обмін значеннями.
3-й байт - адреса елемента пам'яті, з яким виконується обмін значеннями.

Команди арифметичних та логічних операцій з акумулятором

ADDLA | **40h** | 01000000b | 2 bytes | ZF, CF

Додає до числа в акумуляторі вказане значення, та записує результат до акумулятора.
2-й байт - значення, яке потрібно додати до вмісту акумулятора.

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.
Якщо в результаті операції в акумуляторі отримуємо переповнення (значення, яке перевищує FFh), тоді прапор CF = 1, інакше CF = 0.

ADDRA | **41h** | 01000001b | 2 bytes | ZF, CF

Додає вміст акумулятора до числа в елементі пам'яті, що адресується. Записує результат в акумулятор.
2-й байт - адреса элемента пам'яті, вміст якого потрібно додати до вмісту акумулятора.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.
Якщо в результаті операції в акумуляторі отримуємо переповнення (значення, яке перевищує FFh), тоді прапор CF = 1, інакше CF = 0.

SUBLA | **42h** | 01000010b | 2 bytes | ZF, CF

Відняти від вмісту акумулятора AC вказане значення і залишити результат в акумуляторі.
2-й байт - число, яке потрібно відняти від вмісту акумулятора.
Якщо в результаті операції в акумуляторі отримуємо 00h, тоді прапор ZF = 1, інакше ZF = 0
Якщо в результаті операції акумуляторі отримуємо негативне число, тоді прапор CF = 1,

інакше CF = 0.

SUBRA | **43h** | 01000011b | 2 bytes | ZF, CF

Відняти від вмісту акумулятора значення із вказаного елементу пам'яті і залишити результат в акумуляторі.
2-й байт - адреса елемента пам'яті, вміст якого потрібно відняти від вмісту акумулятора.
Якщо в результаті операції в акумуляторі отримуємо 00h, тоді прапор ZF = 1, інакше ZF = 0
Якщо в результаті операції акумуляторі отримуємо негативне число, тоді прапор CF = 1, інакше CF = 0.

ANDLA | **44h** | 01000100b | 2 bytes | ZF

Виконати логічне "AND" з вмістом акумулятора і вказаним значенням. Результат записати до акумулятора.
2-й байт - значення, з яким потрібно зробити логічне "AND" зі вмістом акумулятора.

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.
Якщо в результаті операції в акумуляторі отримуємо переповнення (значення, яке перевищує FFh), тоді прапор CF = 1, інакше CF = 0.

ANDRA | **45h** | 01000101b | 2 bytes | ZF

Виконати логічне "AND" вмісту акумулятора із значенням вказаного елементу пам'яті. Результат записати до акумулятора.
2-й байт - адреса елемента пам'яті, з вмістом якого потрібно зробити операцію "AND".

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.
Якщо в результаті операції в акумуляторі отримуємо переповнення (значення, яке перевищує FFh), тоді прапор CF = 1, інакше CF = 0.

ORLA | **46h** | 01000110b | 2 bytes | ZF

Виконати логічне "OR" вмісту акумулятора і вказаного значення.
2 байт - значення, з яким потрібно зробити логічне "OR".

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

ORRA | **47h** | 01000111b | 2 bytes | ZF

Виконати логічне "OR" вмісту акумулятора із значенням із вказаного елемента пам'яті.
2-й байт - адреса елемента пам'яті, з вмістом якого потрібно зробити логічне "OR".

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

XORLA | **48h** | 01001000b | 2 bytes | ZF

Виконує операцію логічне "XOR" між вмістом акумулятора і вказаним значенням. Результат записується в акумулятор.
2-й байт - значення, з яким потрібно зробити логічне "XOR".

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

XORRA | **49h** | 01001001b | 2 bytes | ZF

Виконати логічне "XOR" вмісту акумулятора із значенням у елементі пам'яті. Результат записується в акумулятор.
2-й байт - адреса елемента пам'яті, зі значенням в якому потрібно зробити логічне "XOR".
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

DECA | **4Ah** | 01001010b | 1 byte | ZF, CF

Зменшення на 1 значення в акумуляторі AC.
Якщо в результаті операції в акумуляторі отримуємо 00h, тоді прапор ZF = 1, інакше ZF = 0
Якщо в результаті операції акумуляторі отримуємо негативне число, тоді прапор CF = 1, інакше CF = 0.

INCA | **4Bh** | 01001011b | 1 byte | ZF, CF

Збільшує значення в акумуляторі AC на 1.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.
Якщо в результаті операції в акумуляторі отримуємо переповнення (значення, яке перевищує FFh), тоді прапор CF = 1, інакше CF = 0.

DAA | **4Ch** | 01001100b | 1 byte | ZF, CF

Десяткова корекція акумулятора після складання двійково-десяткових чисел.

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.
Якщо в результаті операції в акумуляторі отримуємо переповнення (значення, яке перевищує FFh), тоді прапор CF = 1, інакше CF = 0.

DAS | **4Dh** | 01001101b | 1 byte | ZF, CF

Десяткова корекція акумулятора після віднімання двійково-десяткових чисел.

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.
Якщо в результаті операції в акумуляторі отримуємо переповнення (значення, яке перевищує FFh), тоді прапор CF = 1, інакше CF = 0.

NOTA | **4Eh** | 01001110b | 1 byte | ZF

Інверсія вмісту акумулятора.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

AAD | **3Eh** | 00111110b | 1 bytes | ZF

Перетворює число в акумуляторі з двійково-десяткового у двійковий вигляд.
Якщо в результаті операції в Акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

AAA | **3Fh** | 00111111b | 1 bytes | ZF, CF

Перетворює число в Акумуляторі з двійкового у двійково-десятковий вигляд.
Якщо в результаті операції в Акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.
Якщо отримуємо переповнення в Акумуляторі (значення, більше за FFh), прапор переносу CF=1, інакше CF=0.

ADDLACF | **88h** | 10001000b | 2 bytes | ZF, CF

Додає до вмісту Акумулятора буквальне значення, що міститься у другому байті команди та значення біта CF. Записує результат у Акумулятор.
2-й байт - буквальне значення, яке потрібно додати до вмісту Акумулятора.
Якщо в результаті операції в Акумуляторі отримуємо 00h, то прапор ZF = 1, інакше прапор ZF = 0.
Якщо отримуємо переповнення в Акумуляторі (значення, що більше за FFh), прапор переносу CF=1, інакше CF=0.

ADDRACF | **89h** | 10001001b | 2 bytes | ZF, CF

Додає до вмісту Акумулятора значення з адресованого елемента пам'яті та значення біта CF. Записує результат в Акумулятор.
2-й байт - адреса елемента пам'яті, вміст якого потрібно додати до вмісту Акумулятора.
Якщо в результаті операції в Акумуляторі отримуємо 00h, то прапор ZF = 1, інакше прапор ZF = 0.
Якщо отримуємо переповнення в Акумуляторі (значення, що більше за FFh), прапор переносу CF=1, інакше CF=0.

SUBLACF | **8Ah** | 10001010b | 2 bytes | ZF, CF

Відняти від вмісту Акумулятора буквальне значення, та значення біта CF. Результат залишити в Акумуляторі.
2-й байт - буквальне значення, яке потрібно відняти від вмісту Акумулятора.
Якщо в результаті операції в Акумуляторі отримуємо 00h, то прапор ZF = 1, інакше прапор ZF = 0.
Якщо отримуємо переповнення в Акумуляторі (значення, що більше за FFh), прапор переносу CF=1, інакше CF=0.

SUBRACF | **8Bh** | 10001011b | 2 bytes | ZF, CF

Відняти від вмісту Акумулятора значення з елемента пам'яті та біт CF. Результат залишити в Акумуляторі.
2-й байт - адреса елемента пам'яті, вміст якого потрібно відняти від вмісту Акумулятора.
Якщо в результаті операції в Акумуляторі отримуємо 00h, то прапор ZF = 1, інакше прапор ZF = 0.
Якщо отримуємо переповнення в Акумуляторі (значення, що більше за FFh), прапор переносу CF=1, інакше CF=0.

Команди арифметичних та логічних операцій з пам'яттю

DECR | **50h** | 01010000b | 2 bytes

Зменшення на 1 числа в зазначеному елементі пам'яті.
2-й байт - адреса елемента пам'яті, з яким виконується операція.

INCR | **51h** | 01010001b | 2 bytes

Збільшує на 1 значення в зазначеному елементі пам'яті.
2-й байт - адреса елемента пам'яті, з яким виконується операція.

Команди зсуву вмісту акумулятора

SHIFTLA | **60h** | 01100000b | 1 byte | ZF, CF

Зсув вліво даних в акумуляторі. Крайній лівий біт з акумулятора переходить у прапор перенесення CF.

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

SHIFTRA | **61h** | 01100001b | 1 byte | ZF, CF

Зсув вправо даних в акумуляторі. Крайній правий біт із акумулятора переходить у прапор перенесення CF.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

ROLACF | **62h** | 01100010b | 1 byte | ZF, CF

Зсув вліво даних в акумуляторі по колу через біт прапора перенесення CF. Крайній лівий біт з акумулятора переходить у прапор перенесення CF. У молодший біт поміщається попереднє значення прапора перенесення CF.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

RORACF | **63h** | 01100011b | 1 byte | ZF, CF

Зсув вправо даних в акумуляторі по колу через біт прапора перенесення CF. Крайній правий біт з акумулятора відправляється у прапор перенесення CF. У старший біт поміщається попереднє значення прапора CF.

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

Команди зсуву в елементах пам'яти

SHIFTLR | **70h** | 01110000b | 2 bytes

Зсув вліво даних в зазначенному елементі пам'яті. Крайній лівий біт із зазначенного елемента пам'яті втрачається.
2-й байт - адреса елемента пам'яті, з яким виконується операція.

SHIFTRR | **71h** | 01110001b | 2 bytes

Зсув вправо даних в зазначенному елементі пам'яті. Крайній правий біт з зазначенного елемента втрачається.
2-й байт - адреса елемента пам'яті, з яким виконується операція.

Команди для роботи з бітами акумулятора і прапорами

CBA | **80h** | 10000000b | 2 bytes | ZF

Встановити в 0 зазначений біт в акумуляторі.
2-й байт - номер біта, який встановлюється в 0.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

SBA | **81h** | 10000001b | 2 bytes

Встановити в 1 зазначений біт в акумуляторі AC.
2-й байт - номер біта, який встановлюється в 1.

XCHGAA | **82h** | 10000010b | 1 byte | ZF

Міняє місцями старшу і молодшу половини байта в акумуляторі AC.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

CLRCF | **83h** | 10000011b | 1 byte

Скидає біт прапора перенесення CF в 0.

CLRTF | **84h** | 10000100b | 1 byte

Скидає біт прапора TF переповнення лічильника адреси в 0.

MOVCFA | **86h** | 10000110b | 2 bytes | ZF

Скопіювати вміст прапора CF в зазначений біт акумулятора.
2-й байт - номер біта, якому присвоюється значення біта CF.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

MOVACF | **87h** | 10000111b | 2 bytes | ZF, CF

Скопіювати вміст зазначеного біта акумулятора в біт прапора CF.
2-й байт - номер біта в акумуляторі, значення якого записується в біт CF.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

Команди для роботи з бітами в пам'яті

CBR | **90h** | 10010000b | 3 bytes

Встановити в 0 зазначений біт в зазначеному елементі пам'яті.
2-й байт - номер біта, що встановлюється в 0.
3-й байт - адреса елемента пам'яті, з яким виконується операція.

SBR | **91h** | 10010001b | 3 bytes

Встановити в 1 зазначений біт в зазначенному елементі пам'яті.
2-й байт - номер біта, який встановлюється в 1.

3-й байт - адреса елемента пам'яті, з яким виконується операція.

MOVCFR | **92h** | 10010010b | 3 bytes

Скопіювати вміст прапора CF в зазначений біт елемента пам'яті.

2-й байт - номер біта, в який копіюється значення CF.
3-й байт - адреса елемента пам'яті, з яким проводиться операція.

MOVRCF | **93h** | 10010011b | 3 bytes | CF

Скопіювати вміст зазначеного біта пам'яті в біт прапора CF.

2-й байт - номер біта, значення якого копіюється в CF.
3-й байт - адреса елемента пам'яті, з яким проводиться операція.

Команди для роботи зі стеком

PUSHA | **A0h** | 10100000b | 1 byte | ZF

Запис в стек вмісту акумулятора AC.

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

PUSHR | **A1h** | 10100001b | 2 bytes

Запис в стек вмісту зазначеного елемента пам'яті.
2-й байт - адреса елемента пам'яті, значення з якого записується в стек.

PUSHL | **A2h** | 10100010b | 2 bytes

Запис байта в стек.
2-й байт - значення, яке буде занесено в стек.

POPA | **A3h** | 10100011b | 1 byte | ZF

Перенести значення зі стека в акумулятор AC.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

POPR | **A4h** | 10100100b | 2 bytes

Перенести значення зі стека в зазначений елемент пам'яті.

2-й байт - адреса елемента пам'яті, в який записуються дані зі стека.

MOVSPA | **A5h** | 10100101b | 1 byte | ZF

Занести вміст покажчика стека SP в акумулятор AC.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

MOVASP | **A6h** | 10100110b | 1 byte | ZF

Записати вміст акумулятора в покажчик стека SP.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

SETSP | **A7h** | 10100111b | 2 bytes

Записати байт в покажчик стека SP.
2-й байт - дані, які будуть записані в покажчик стека.

INITSP | **A8h** | 10101000b | 1 byte

Записує фіксоване початкове значення 251 (11111011b) в покажчик стека SP.

Команди для роботи з підпрограмами та безумовний перехід

CALL | **B0h** | 10110000b | 2 bytes

Викликає виконання підпрограми до тих пір, поки не зустріне команду **RETURN**. Після чого повертається з підпрограми і починає виконувати наступну команду.
2-й байт - адреса входу в підпрограму.

RETURN | **B1h** | 10110001b | 1 byte

Команда повернення з підпрограми, яку було викликано командою **CALL**.

JMP | **B2h** | 10110010b | 2 bytes

Безумовний перехід. Перейти до виконання команди, починаючи із зазначеної адреси.
2-й байт - адреса переходу, з якого буде продовжено виконання програми.

Команди умовних переходів

LOOP | **C0h** | 11000000b | 3 bytes

Умовний перехід з лічильником. Зменшує вміст зазначеного елементу пам'яті на одиницю. Якщо результат не дорівнює нулю, переходить за адресою, вказаною у 3-му байті команди. Інакше виконує наступну команду.

2-й байт - адреса елемента пам'яті, вміст якого зменшується на одиницю.
3-й байт - адреса переходу, якщо після віднімання результат не дорівнює нулю.

LOOPI | **C1h** | 11000001b | 3 bytes

Умовний перехід з лічильником. Збільшує вміст зазначеного елемента пам'яті на одиницю. Якщо результат не дорівнює нулю, переходить за адресою, вказаною в 3-му байті команди. Інакше виконує наступну команду.

2-й байт - адреса елемента пам'яті, вміст якого збільшується на одиницю.
3-й байт - адреса переходу, якщо після складання результат не дорівнює нулю.

JRBNZ | **C2h** | 11000010b | 4 bytes

Перевірити зазначений біт за вказаною адресою. Якщо цей біт = 0, то почати виконувати наступну команду. Якщо він = 1, перейти за адресою, яку зазначено в 4-му байті команди.

2-й байт - номер біта, який перевіряється.
3-й байт - адреса байта, в якому перевіряється біт.
4-й байт - адреса переходу, якщо біт = 1.

JRBZ | **C3h** | 11000011b | 4 bytes

Перевірити зазначений біт за вказаною адресою. Якщо цей біт = 1, то почати виконувати наступну команду. Якщо він = 0, перейти за адресою, яку зазначено в 4-му байті команди.

2-й байт - номер біта, який перевіряється.
3-й байт - адреса байта, в якому перевіряється біт.
4-й байт - адреса переходу, якщо біт = 0.

JZFNZ | **C4h** | 11000100b | 2 bytes

Перевірити біт прапора нуля ZF. Якщо ZF = 1, перейти за вказаною адресою переходу. Якщо ZF = 0, виконати наступну команду.
2-й байт - адреса переходу, якщо прапор ZF = 1.

JZFZ | **C5h** | 11000101b | 2 bytes

Перевірити біт прапора нуля ZF. Якщо ZF = 0, перейти за вказаною адресою переходу. Якщо ZF = 1, виконати наступну команду.
2-й байт - адреса переходу, якщо прапор ZF = 0.

JCFNZ | **C6h** | 11000110b | 2 bytes

Перевірити біт прапора перенесення CF. Якщо CF = 1, перейти за вказаною адресою переходу. Якщо CF = 0, виконати наступну команду.

2-й байт - адреса переходу, якщо прапор CF = 1.

JCFZ | **C7h** | 11000111b | 2 bytes

Перевірити біт прапора перенесення CF. Якщо CF = 0, перейти за вказаною адресою переходу. Якщо CF = 1, виконати наступну команду.

JTFNZ | **C8h** | 11001000b | 2 bytes

Перевірити біт прапора TF. Якщо TF = 1, перейти за вказаною адресою переходу. Якщо TF = 0, виконати наступну команду.

2-й байт - адреса переходу, якщо прапор TF = 1.

JTFZ | **C9h** | 11001001b | 2 bytes

Перевірити біт прапора TF. Якщо TF = 0, перейти за вказаною адресою переходу. Якщо TF = 1, виконати наступну команду.

2-й байт - адреса переходу, якщо прапор TF = 0.

JALR | **B7h** | 10110111b | 3 bytes

Порівняти вміст Акумулятора зі вмістом елемента пам'яті. Якщо значення в Акумуляторі менше за значення в адресованому елементі пам'яті, то перейти за адресою, яку вказано у третьому байті команди.
2-й байт - адреса елемента пам'яті, в якому знаходиться число, що порівнюється.
3-й байт - адреса переходу.

JALL | **B8h** | 10111000b | 3 bytes

Порівняти вміст Акумулятора із буквальним значенням. Якщо значення в Акумуляторі менше за буквальне значення, то перейти за адресою, яку вказано у третьому байті команди.

2-й байт - буквальне значення, що порівнюється.
3-й байт - адреса переходу.

JAER | **B9h** | 10111001b | 3 bytes

Порівняти вміст Акумулятора зі вмістом елемента пам'яті. Якщо значення в Акумуляторі дорівнює значенню в елементі пам'яті, то перейти за адресою, яку вказано у третьому байті команди.
2-й байт - адреса елемента пам'яті, у якому знаходиться число, що порівнюється.
3-й байт - адреса переходу.

JAEL | **BAh** | 10111010b | 3 bytes

Порівняти вміст Акумулятора із буквальним значенням. Якщо значення в Акумуляторі дорівнює буквальному значенню, то перейти за адресою, яку вказано у третьому байті команди.
2-й байт - буквальне значення, що порівнюється.
3-й байт - адреса переходу.

JAGR | **BBh** | 10111011b | 3 bytes

Порівняти вміст Акумулятора із вмістом адресованого елемента пам'яті. Якщо значення в Акумуляторі більше ніж значення у елементі пам'яті, то перейти за адресою, яку вказано в третьому байті команди.
2-й байт - адреса елемента пам'яті, у якому знаходиться число, що порівнюється.
3-й байт - адреса переходу.

JAGL | **BCh** | 10111100b | 3 bytes

Порівняти вміст Акумулятора із буквальним значенням. Якщо значення в Акумуляторі більше за буквальне значення, то перейти за адресою, яку вказано у третьому байті команди.
2-й байт - буквальне значення, що порівнюється.
3-й байт - адреса переходу.

JRLR | **BDh** | 10111101b | 3 bytes

Порівняти вміст одного елемента пам'яті зі вмістом другого елемента пам'яті. Якщо значення в першому елементі менше ніж значення в другому елементі пам'яті, то перейти за адресою, яку вказано в четвертому байті команди.
2-й байт - адреса елемента пам'яті, у якому знаходиться перше число, що порівнюється.
3-й байт - адреса елемента пам'яті, у якому знаходиться друге число, що порівнюється.
4-й байт - адреса переходу.

JRER | **BEh** | 10111110b | 4 bytes

Порівняти вміст одного елемента пам'яті зі вмістом другого елемента пам'яті. Якщо значення в першому елементі дорівнює значенню в другому елементі пам'яті, то перейти за адресою, яку вказано в четвертому байті команди.
2-й байт - адреса елемента пам'яті, у якому знаходиться перше число, що порівнюється.
3-й байт - адреса елемента пам'яті, у якому знаходиться друге число, що порівнюється.
4-й байт - адреса переходу.

JRGER | **BFh** | 10111111b | 4 bytes

Порівняти вміст одного елемента пам'яті зі вмістом другого елемента пам'яті. Якщо значення в першому елементі дорівнює або більше ніж значення в другому елементі пам'яті, то перейти за адресою, яку вказано в четвертому байті команди.
2-й байт - адреса елемента пам'яті, у якому знаходиться перше число, що порівнюється.
3-й байт - адреса елемента пам'яті, у якому знаходиться друге число, що порівнюється.
4-й байт - адреса переходу.

Команди введення / виведення

OUTDO | **D0h** | 11010000b | 1 byte

Записує значення з акумулятора AC в регістр DO (елемент пам'яті 255).

INDI | **D1h** | 11010001b | 1 byte | ZF

Записує значення з регістра вхідних даних DI (елемент за адресою 253) в акумулятор AC.
Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

INKBD | **D2h** | 11010010b | 1 byte | ZF

Записує в аккумулятор код клавіші що була натиснута останньою. Призупиняє виконання програми і вмикає сигнал ● **HALT** жовтого кольору до натискання будь-якої клавіші.

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

OUTKBD | **D3h** | 11010011b | 1 byte | ZF

Записує байт акумулятора в порт управління кольором клавіш. Молодші 6 біт визначають адресу клавіші, старші 2 біта керують кольором клавіші.

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

OUTCLRKBD | **D4h** | 11010100b | 1 bytes

Вимикає підсвічування всіх клавіш (цифри + алфавіт), крім функціональних.

INCOLKBD | **D5h** | 11010101b | 1 bytes

Повертає в Акумулятор колір клавіші. Перед викликом команди код клавіші записується в 6 молодших біт Акумулятора. Після виконання команди код кольору повертається в два старших біти Акумулятора.

Розширені команди (ланцюгові операції і множення)

MOVSTR | **E0h** | 11100000b | 4 bytes | TF

Копіює задану кількість байт з пам'яті, починаючи з "адреси-джерела", в пам'ять, починаючи з "адреси-одержувача".
2-й байт - кількість байт, що буде скопійовано.
3-й байт - адреса першого елемента пам'яті в масиві, з якого буде прочитано дані.
4-й байт - адреса першого елемента пам'яті в масиві, в який буде записано дані.

Якщо в результаті виконання команди значення адреси елемента пам'яті виходить за межі адресного простору, тоді прапор TF = 1, інакше TF = 0.

MULRA | **E1h** | 11100001b | 3 bytes

Помножити значення в акумуляторі на значення в зазначеному елементі пам'яті. Третій байт - адреса елемента пам'яті, в якому буде розміщений молодший байт результату. Старший байт результату розміщується за адресою, на 1 більшою від адреси молодшого.
2-й байт - адреса елемента пам'яті, в якому знаходиться другий множник.
3-й байт - адреса елемента пам'яті, в якому буде розміщено молодший байт результату.

Якщо в результаті операції в акумуляторі отримуємо 00h, то прапор ZF = 1, інакше ZF = 0.

DIVRA | **E2h** | 11100010b | 3 bytes | ZF | CF

Поділити значення в зазначенному єлементі пам'яті на число в акумуляторі.
2-й байт - адреса елемента пам'яті, в якому знаходиться молодший байт 16-бітного числа, яке ми ділимо на вміст акумулятора. Старший байт знаходиться за ним.
3-й байт - адреса елемента пам'яті, в якому знаходиться молодший байт результату. Старший байт результату розміщується за адресою, на 1 більшою від адреси молодшого. Дробна частина (залишок від ділення) буде знаходитись за адресою, що на 2 більше від молодшого байта результату.

При діленні на нуль або переповненні 4-й біт регістра прапорів FR встановлюється в 1. Інакше він = 0.

Якщо в акумуляторі знаходиться значення 00h, тоді прапор ZF = 1, інакше ZF = 0.
Якщо в результаті операції отримуємо переповнення (значення, яке перевищує FFFFh), тоді прапор CF = 1, інакше CF = 0.

Рекомендується замість команди **DIVRA** використовувати команду множення **MULRA** на число, що є зворотнім знаменнику.

RETAD | **E3h** | 11100011b | 2 bytes | TF

Мається на увазі, що за цією командою завжди йде двобайтова команда JMP. Запам'ятовує в елементі пам'яті так звану "адресу повернення". Для обчислення "адреси повернення" команда "дізнається" адресу свого першого байта та додає до нього 4.
2-й байт - адреса елементу пам'яті, до якого буде записана "адреса повернення".
Якщо при виконанні цієї команди адреса повернення більше за FFh, то прапор TF=1, інакше TF=0.

X | **E6h** | 11100110b | 2 bytes

Виконує одну команду, адресу якої задано другим байтом команди, після чого продовжує виконання програми.
2-й байт - адреса першого байту команди, що має бути виконана.
Увага! У разі виклика цією командою інструкцій умовних та безумовних переходів, а також самої себе, коректна робота команди не гарантується.