Розділ 2. Команди МП 8088/86

2.1. Склад команд

Мікропроцесор має 92 **команди**, які можна поділити на 7 **груп**:

1. Команди **пересилання** даних між регістрами: комірками і портами введення/виведення;
2. **Арифметичні** команди;
3. Команди над **бітами**, які здійснюють зміщення і логічні операції;
4. Команди **передачі керування**, виклику процедур і повернення з процедури;
5. Команди **обробки рядків**;
6. Команди **переривання** для обробки специфічних подій;
7. Команди **управління процесором** - встановлення і скидання прапорців стану, зміни режиму функціонування МП.

2.1.1. Команди пересилання даних

Команда **MOV** – найчастіше вживається в програмі. Її можна застосовувати для пересилання:

```
MOV AX, CX ; з регістра в регістр
MOV AX, TABLE ; з пам'яті в регістр
MOV TABLE, AX ; з регістра в пам'ять
MOV DS, AX ; з регістра в регістр сегмента
MOV AH, AL ; з регістра байту в регістр
MOV AX, -40 ; константу в регістр
MOV BETA, 02Fh ; константу в пам'ять
```

НЕ МОЖНА виконувати такі пересилання:

1. ***з пам'яті в пам'ять***. Як зазначалось, в двоадресних командах не можна використовувати пряму адресацію в двох операндах. Тому, пересилання пам'ять-пам'ять можна виконувати двома командами:

~~MOV ALPHA, BETA~~

```
MOV AX, BETA
MOV ALPHA, AX
```

2. Вміст **комірок пам'яті** не можна пересилати безпосередньо **в регістр сегменту**, а лише через регістр загального призначення:

~~MOV DS, BETA~~

```
MOV AX, BETA
MOV DS, AX
```

Не можна пересилати дані з одного **регістра сегменту** в інший **регістр сегменту**, а лише через регістр загального призначення :

```
MOV AX, DS
MOV ES, AX
```

~~MOV ES, DS~~

Стек автоматично створюється для роботи з підпрограмами. Але оскільки в МП лише 4 регістра загального призначення, то часто доводиться запам'ятовувати вміст регістрів, щоб звільнити регістри і виконати інші дії. Саме для цього використовуються команди:

PUSH джерело; заслати в стек

POP приймач ; зчитати зі стеку

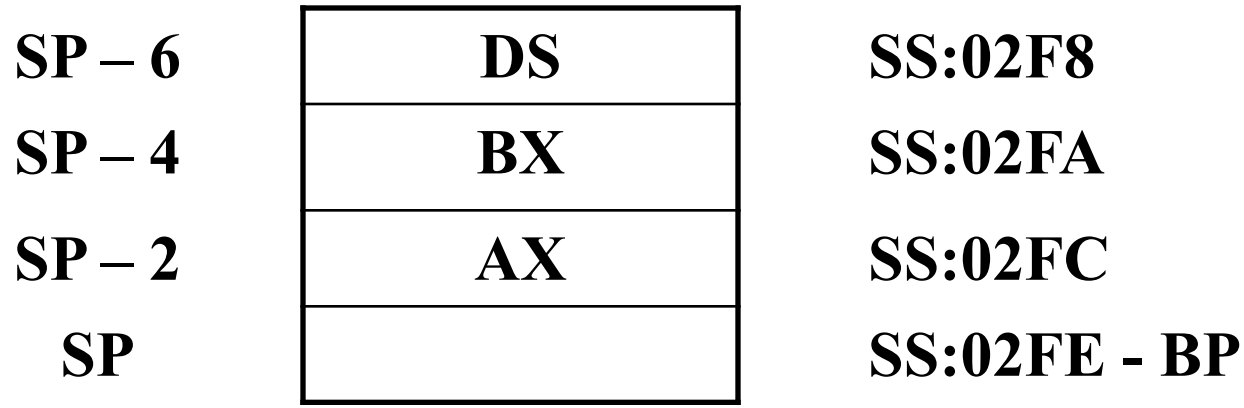
Наприклад:

```
PUSH SI
PUSH DS
PUSH CX
PUSH ALPHA
PUSH DELTA [BI + SI]
```

Дані засилаються на вершину стека, тому при їх зчитуванні необхідно додержуватись відповідної послідовності.

Наприклад:

```
PUSH AX
PUSH BX
PUSH DS
```



Оскільки зверху знаходиться **DS**, а нижче - **BX** і **AX**, то відновлення регістрів потрібно здійснити в протилежному порядку:

```
POP DS
POP BX
POP AX
```

Команди PUSH-POP можна використовувати для обміну між сегментними регістрами.
При цьому не використовуються регістрова пам’ять
Але виконання команд буде довшим:
пара **PUSH-POP реалізується за 26 тактів, а дві команди MOV за 4 такти.**

```
PUSH DS
POP ES
```

26 тактов

```
MOV AX, DS
MOV ES, AX
```

4 такта

2.1.2. Команда обміну XCHG

Назва походить від англійського слова exchange - обміняти. Використовується для обміну вмістом двох регістрів або регістра і пам'яті.

Не МОЖНА використовувати для обміну між *сегментними регістрами.*

~~XCHG CS, DS~~

```
XCHG BX, AX
XCHG AH, BL
XCHG AX, TABLE
XCHG TABLE, AX
```

2.1.3. Команди обміну з портами

IN акумулятор, порт
OUT порт, акумулятор

Акумулятор – це регістр **AX** при обміні **словами** і **AL** при обміні **байтами**. Порт визначається своїм номером від 0 до 256. Можна безпосередньо відзначати номер порту або дати йому ім'я.

```
PORT_NUM EQU 210
IN AX, 200
IN AL, PORT_NUM
OUT DX, AX
OUT 200, AL
```

Також номер порту можна записати в регістр **DX**.

2.1.4. Команда LEA – завантаження ефективної адреси

LEA(load effective address – завантажити виконавчу адресу) пересилає зміщення комірки пам'яті в:
1. певний 16-бітовий регістр ЗП;
2. регістр покажчика;
3. індексний регістр;

LEA регістр 16, пам'ять 16

На відміну від команди **MOV** з операцією **OFFSET**, операнд **пам'ять 16** може бути індексованим, що забезпечує гнучкість адресації.

Наприклад:

LEA BP, TABLE [DI]

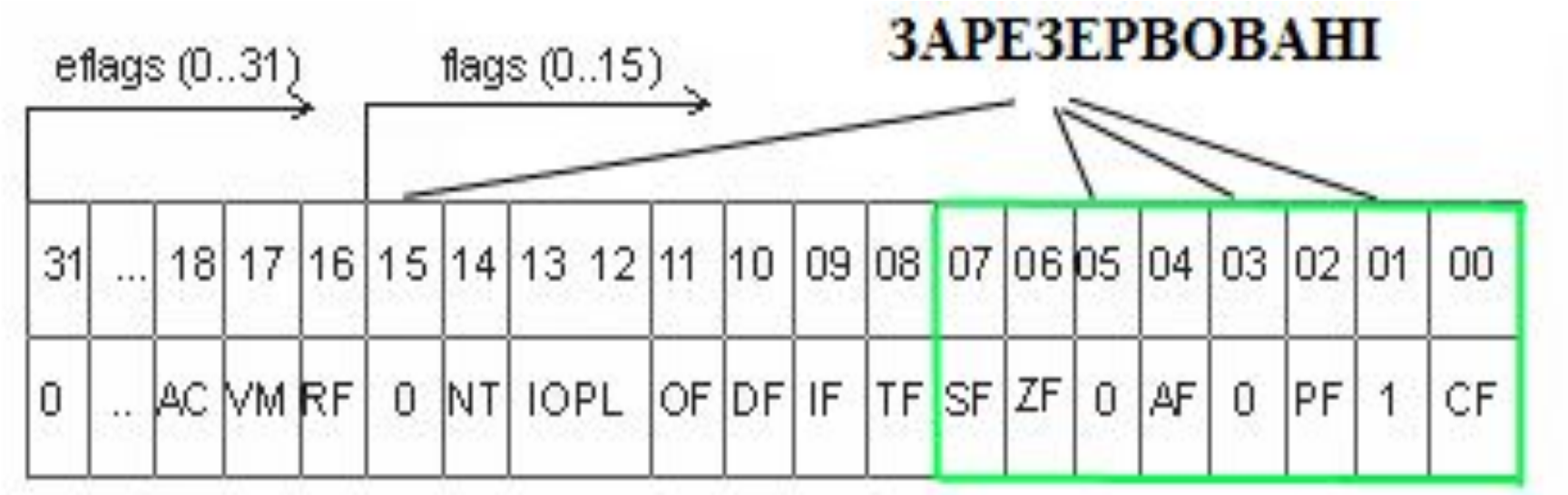
~~MOV BP, offset TABLE [DI]~~

Якщо в **DI** міститься 8, то в **BP** буде заслана адреса **TABLE + 8**.

2.1.5. Команди пересилання прапорців

Можна пересилати в регістр **AH** молодший байт регістра прапорців **F** командою **LAHF**;

LAHF; *F→AH*



Зворотнє пересилання з **AH** в молодший байт регістра **F - SAHF:**

```
SAHF; AH→F
```

Вміст регістру прапорців можна також пересилати в стек командою **PUSHF**, а в зворотному напрямку - **POPF**. Це потрібно для захисту регістра прапорців від зміни при виклику процедур. Якщо немає впевненості, що процедура не змінить регістр прапорців, то його потрібно захистити (зберегти).

Приклад:

```
PUSH AX
PUSH DI
PUSHF
CALL SORT
POPF
POP DI
POP AX
```