



# Лекция 17. Форматы команд процессора

---

- **Формат команды** - это структура кода команды, понимаемая процессором.  
Рассмотрим принципы построения кода команд на примере системы команд x86 для 16-разрядных процессоров 8086 (для простоты). Это базовая система команд для всех процессоров
- Форматы команд можно разделить на 4 группы:
  - форматы команд **без операндов** (или операнды **по умолчанию**),
  - форматы команд **с операндами**
  - форматы команд **прямых переходов и вызовов**
  - **специфические** форматы

## Обозначения полей в байтах команды

---

- **cop** - фиксированный код операции
- **mod** - используется при адресации операнда в памяти
- **reg** - код регистра (**sr** - для сегментного регистра)
- **r/m** - код регистра или способа адресации памяти
- **disp** - поле для размещения числовой величины при указании внутрисегментного адреса операнда:
  - disp-h** – ст.байт поля, **disp-l** – младший байт
- **data** - непосредственный операнд:
  - data-h**- ст.байт, **data-l** -мл. байт
-  - обязательные байты в формате команды
-  - возможные байты

# Формат команд без операндов ( или операнды по умолчанию)

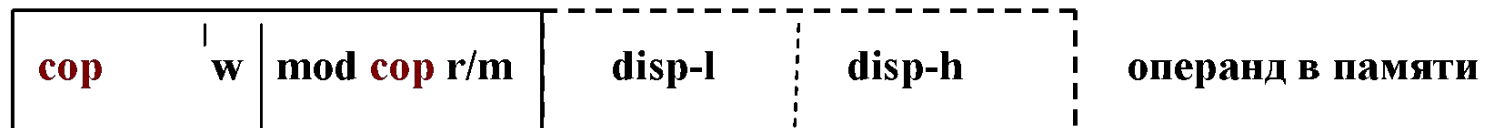
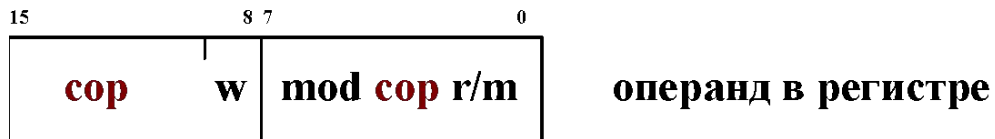
---



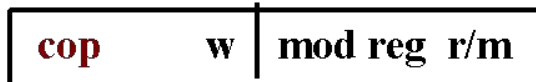
**код операции**

# Форматы команд с одним операндом

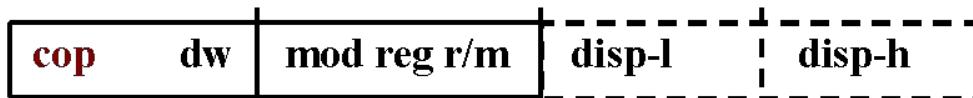
---



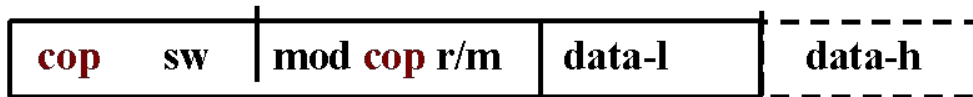
## Форматы команд с двумя операндами



регистр - регистр



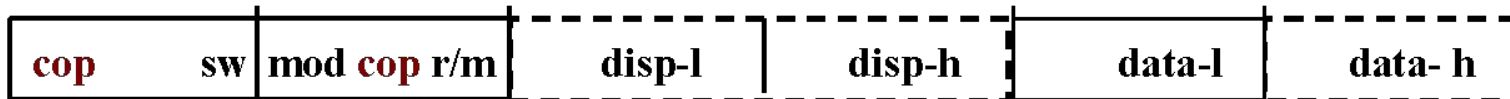
регистр – память  
память - регистр



регистр –  
непосредственный операнд



регистр AX/AL - непосредственный операнд



память – непосредственный операнд

## 1-й байт команды: содержит код операции (COP)

Фиксированный код, который определяет саму команду.

Байт COP может еще содержать **битовые признаки** для процессора:

- **w** – бит разрядности операндов: 0 – байты, 1 - слова
- **d** – местонахождение результата: 0 - в памяти; 1 - в регистре.
- **s** – говорит процессору о необходимости **расширить со знаком** однобайтный непосредственный операнд до формата слова:  
0 - расширение не нужно, 1 - расширение нужно

Пример: какими будут биты **w** , **d** , **s** в машинном коде команд?

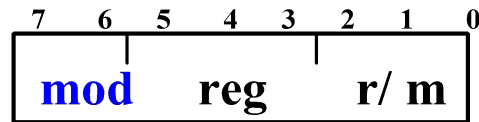
```
add bx, ds:adres    ; w=1 , d= 1
add ds:adres, bl    ; w=0 , d= 0
add dx, 3           ; w=1 , s= 1
add dl, 3           ; w= 0 , s= 0
add dx, 3f1h        ; w= 1 , s= 0
```

## 2-й байт : байт режима адресации

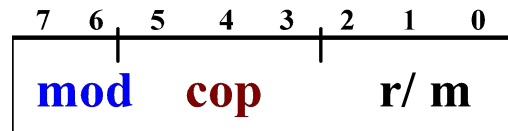
---

Имеет два возможных формата:

- 1) в **2-операндных** командах типа: регистр–регистр, регистр–память или память – регистр



- 2) в командах **с одним операндом** (регистр/память) и в командах **с непосредственным операндом**



Поле reg за «ненужностью» заполнено фиксированным кодом, который называют «вторичный COP»

---

## Поле Mod

Дает процессору информацию о месте размещения операндов в команде.

mod=11: операнда в памяти нет

В этом случае поля **reg**, **sr**, **r/m** содержат **коды регистров**, где размещены операнды:

- при  $w = 0$  - это коды 8- разрядных регистров
- при  $w = 1$  - это коды 16- разрядных регистров

Коды регистров  
в полях **reg** и **r/m**

reg (r/m)	w = 0	w = 1
0 0 0	AL	AX
0 0 1	CL	CX
0 1 0	DL	DX
0 1 1	BL	BX
1 0 0	AH	SP
1 0 1	CH	BP
1 1 0	DH	SI
1 1 1	BH	DI

sr	Сегментный регистр
00	ES
01	CS
10	SS
11	DS



mod ≠ 11 : означает, что **есть операнд в памяти**.

Тогда, поле **r/m** содержит **код способа внутрисегментной адресации этого операнда**.

---

Коды способов адресации в памяти

<b>r/ m</b>	<b>Способ адресации операнда</b>	
<b>000</b>	[ <b>BX</b> + <b>SI</b> + <b>disp</b> ]	
<b>001</b>	[ <b>BX</b> + <b>DI</b> + <b>disp</b> ]	
<b>010</b>	[ <b>BP</b> + <b>SI</b> + <b>disp</b> ]	
<b>011</b>	[ <b>BP</b> + <b>DI</b> + <b>disp</b> ]	
<b>100</b>	[ <b>SI</b> + <b>disp</b> ]	
<b>101</b>	[ <b>DI</b> + <b>disp</b> ]	
<b>110</b>	при <b>mod= 00</b>	при <b>mod= 01,10</b>
	Прямой адрес [disp]	[ <b>BP</b> + <b>disp</b> ]
<b>111</b>	[ <b>BX</b> + <b>disp</b> ]	

**!!** Внимание:

Код **r/m=110** кодирует два способа адресации. Для их отличия используется **mod**:

- при **r/m=110** и **mod=00** – это «прямая адресация» и прямой адрес всегда занимает 2-х байтное поле **disp** (т.к. 16-разрядная внутрисегм.адресация)
  - при **r/m=110** и **mod=01** или **10** – это косвенная адресация вида [ **BP+disp** ]
-

## Поле Disp

---

- При **прямой адресации**: поле `disp` содержит 16-разрядный прямой внутрисегментный адрес. То есть поле всегда двухбайтное.
- При **косвенной адресации**: поле `disp` используется для размещения числового значения смещения при задании косвенного адреса. Например: `[bx-1]`, `[si+bx+23Fh]`

Конкретную длину поля `disp` в команде в этом случае процессор определяет по значению битов поля `mod`:

- `mod = 0 0` - `disp` в команде отсутствует
- `mod = 0 1` - `disp` занимает 1 байт
- `mod = 1 0` - `disp` занимает 2 байта

## Адресация операнда в памяти: «префикс сегмента»

---

- Если при задании адреса операнда используется **указатель сегмента**, отличный от DS, в формате команды перед байтом СОР должен находиться **байт «префикс сегмента»**.

Например: `add bl, es: [si]`

Формат байта «префикса сегмента»:

sr – код сегментного регистра

0 0 1 <b>sr</b> 1 1 0
-----------------------

Пример. Создать машинный код команды ADD AX, ds:[SI-2]. Дать подробные пояснения

1. Определимся с форматом команды. Команда с операндами «регистр – память». Следовательно, ее формат может быть такой (от 2 до 4 байтов):



2. Смысл и значения битов в необходимых полях машинного кода:

cop - код операции. Код ADD = 000000 (из Таблицы кодов операций)

бит d - кто первый операнд в команде. Регистр. Следовательно =1

бит w – разрядность операндов. Слова. Следовательно =1

mod (2 бита) – есть ли операнд в памяти и нужно ли поле disp. Операнд в памяти есть, его адрес задается косвенно с числовым смещением  $-2_{10}$ . Для размещения  $-2_{10}$  в команде достаточно 1-байта поля disp. Следовательно = 01

reg (3 бита) – код регистра. Код AX=000 (из Таблицы кодов регистров)

r/m (3 бита) - код способа адресации операнда в памяти. Косвенная адресация вида [SI+disp], ее код =100 (из Таблицы кодов способов адресации)

disp-l – байт для размещения  $-2_{10}$ . Значения битов =11111110

3. Итог. Машинный код 3-байтной команды:

В bin: 00000011 01000100 11111110

В hex: **03 44 FE**

Занесем машинный код команды 03 44 FE в кодовый сегмент «руками» в Отладчике побайтно через нижнее окно отображения памяти, начиная с адреса CS:010E.

Если наш машинный код правильный, Отладчик правильно покажет эту команду в символическом виде в окне кодового сегмента

The screenshot shows a debugger interface with two windows. The top window displays the disassembly of the code segment starting at CS:0100. The bottom window shows the memory editing process at address ds:010E.

```
File Edit View Run Breakpoints Data Options Window Help
[ ]-CPU 80486
cs:0100 0000 add [bx+si],al ax 0000
cs:0102 0000 add [bx+si],al bx 0000
cs:0104 0000 add [bx+si],al cx 0000
cs:0106 0000 add [bx+si],al dx 0000
cs:0108 0000 add [bx+si],al si 0000
cs:010A 0000 add [bx+si],al di 0000
cs:010C 0000 add [bx+si],al bp 0000
cs:010E 0000 add [bx+si],al sp 0080
cs:0110 0000 add [bx+si],al ds 5293
cs:0112 0000 add [bx+si],al es 5293
cs:0114 0000 add [bx+si],al ss 5293
cs:0116 0000 add [bx+si],al cs 5293
cs:0118 0000 add [bx+si],al ip 0100
cs:011A 0000 add [bx+si],al
cs:011C 0000 add [bx+si],al

ds:010E 00 00 00 0
ds:0116 00 00 00 0
ds:011E 00 00 00 0
```

Enter new data [ ]-CPU 80486

```
03 44 0feh
OK Clip... Cancel
```

```
[ ]-CPU 80486
cs:0100 0000 add [bx+si],al
cs:0102 0000 add [bx+si],al
cs:0104 0000 add [bx+si],al
cs:0106 0000 add [bx+si],al
cs:0108 0000 add [bx+si],al
cs:010A 0000 add [bx+si],al
cs:010C 0000 add [bx+si],al
cs:010E 0344FE add ax,[si-02]
cs:0111 0000 add [bx+si],al
cs:0113 0000 add [bx+si],al
cs:0115 0000 add [bx+si],al
cs:0117 0000 add [bx+si],al
cs:0119 0000 add [bx+si],al
cs:011B 0000 add [bx+si],al
cs:011D 0000 add [bx+si],al

ds:010E 03 44 FE 00 00 00 00 00 00
ds:0116 00 00 00 00 00 00 00 00 00
```

## Создать машинный код команды: sub bx, -4

1. Определимся с форматом команды. Команда с операндами «регистр – непосредственный операнд».

Следовательно, ее формат может быть такой (от 2 до 4 байтов):



2. Смысл и значения битов в необходимых полях машинного кода:

cop - код операции. Код команды sub = 100000 (из Таблицы кодов операций)

бит s - надо ли расширять непоср.операнд ?Надо, следовательно =1

бит w – разрядность операндов. Слова. Следовательно =1

mod (2 бита) – есть ли операнд в памяти и нужно ли поле disp?. Операнда в памяти в команде НЕТ , disp не нужен. Следовательно mod= 11

cop (3 бита) – вторичный коп= 101 (см. Таблицу кодов операций)

r/m (3 бита) – код регистра. Код ВХ=011 (из Таблицы кодов регистров)

date-l – непосредственный операнд (-4). Значения битов =11111100

3. Итог. Машинный код 3-байтной команды:

В bin: 10000011 11101 011 11111100

В hex: **83 EB FC**