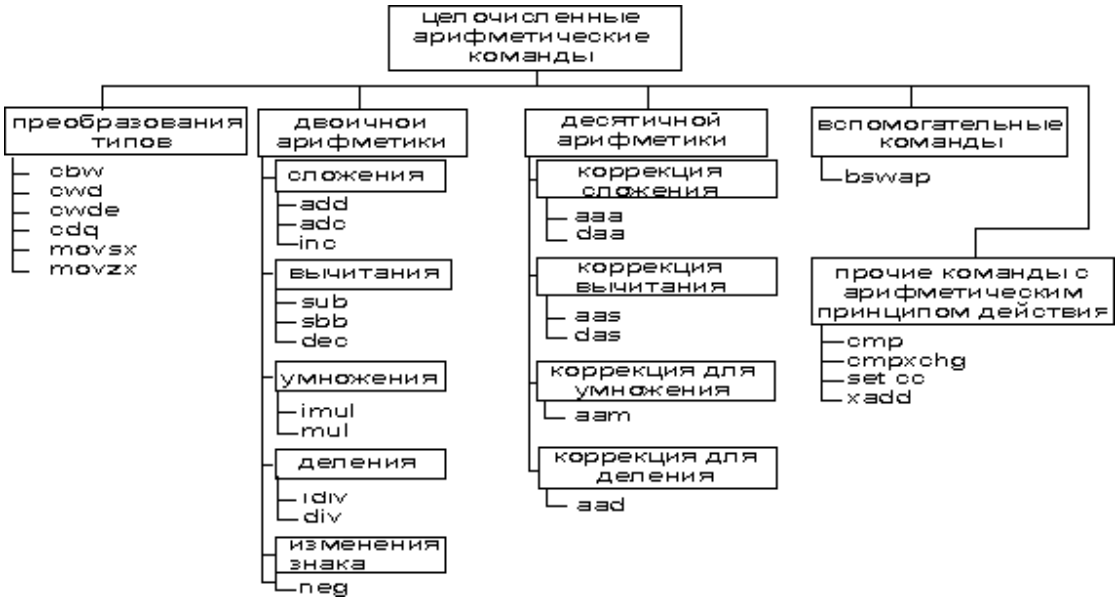
Лекция 7

2.2 Арифметические команды

Целочисленное вычислительное устройство поддерживает чуть больше десятка арифметических команд.



В МП 8088/86 есть группа команд для реализации 4 арифметических действий (добавление, вычитание, умножение, деление) над целыми числами. При этом число может быть записано как в байт, так и в слово. Содержание байта или слова можно рассматривать как число со знаком, или без него. Некоторые разновидности этих команд позволяют реализовать арифметические действия для чисел повышенной точности. Кроме этого, с помощью обычных арифметических операций можно реализовать действия над числами в ВСД-формате. Ясно, что результат будет правильным. Однако его можно скорректировать. Поэтому предусмотрено для такого случая команды коррекции результатов. Их рассматривать не будем. Напомним, что большинство двухоперандных команд действуют так, что изменяется операнд-приемник, а операнд-источник остается неизменным.

1 — после выполнения команды флаг устанавливается (равен 1);
0 — после выполнения команды флаг сбрасывается (равен 0);
r — значение флага зависит от результата работы команды;
? — после выполнения команды флаг не определен;
пробел — после выполнения команды флаг не изменяется;

2.2.1. Команды добавления

Существует три команды:
ADD (add -- прибавить);
ADC (add with carry) - прибавить с переносом;
INC (increment) - прибавить единицу.

ADD операнд_1, операнд_2 — команда сложения с принципом действия:
операнд_1 = операнд_1 + операнд_2

```
ADD AX, CX              AX + CX → AX
```

В этой команде могут наводиться 2 регистра общего назначения, один операнд слово памяти (со всеми возможными режимами адресации) и непосредственные данные как источник.

```
ADD AX, ALPHA
ADD ALPHA, AX
ADD AH, 20
ADD ALPHA, 30
```

Эта команда влияет на 6 флагов.

11	07	06	04	02	00
OF	SF	ZF	AF	PF	CF
r	r	r	r	r	r

ADC операнд_1, операнд_2 — команда сложения с учетом флага переноса **cf**.
Принцип действия команды:
операнд_1 = операнд_1 + операнд_2 + значение_cf

11	07	06	04	02	00
OF	SF	ZF	AF	PF	CF
r	r	r	r	r	r

INC операнд - увеличение значения операнда в памяти или регистре на 1

11	07	06	04	02
OF	SF	ZF	AF	PF
r	r	r	r	r

2.2.2. Команды вычитания

Аналогично к командам добавления, существуют две команды вычитания:
SUB (substract) – вычесть;
SBB (substract with borrow) - вычесть с заемом;
DEC (DECrement) - вычесть единицу.

SUB операнд_1, операнд_2 - целочисленное вычитание
операнд_1=операнд_1-операнд_2

11	07	06	04	02	00
OF	SF	ZF	AF	PF	CF
r	r	r	r	r	r

SBB операнд_1, операнд_2 - целочисленное вычитание с учетом результата предыдущего вычитания командами sbb и sub (по состоянию флага переноса **cf**):

1. выполнить сложение операнд_2 = операнд_2+(**cf**);
2. выполнить вычитание операнд_1 = операнд_1-операнд_2;

11	07	06	04	02	00
OF	SF	ZF	AF	PF	CF
r	r	r	r	r	r

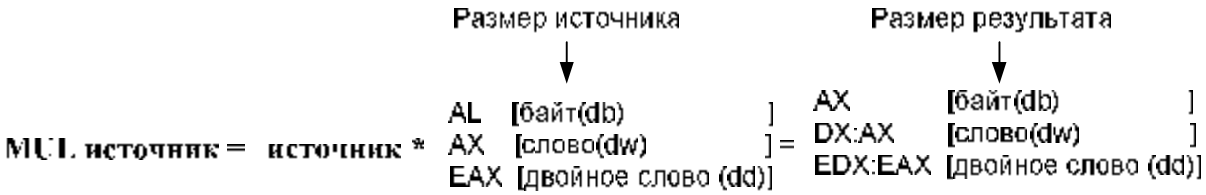
DEC операнд - - уменьшение значения операнда в памяти или регистре на 1

11	07	06	04	02
OF	SF	ZF	AF	PF
r	r	r	r	r

2.2.3. Команды умножения

Существует две команды:
MUL (multiply) - умножение без знака;
IMUL (integer multiply) - умножение со знаком.

MUL источник



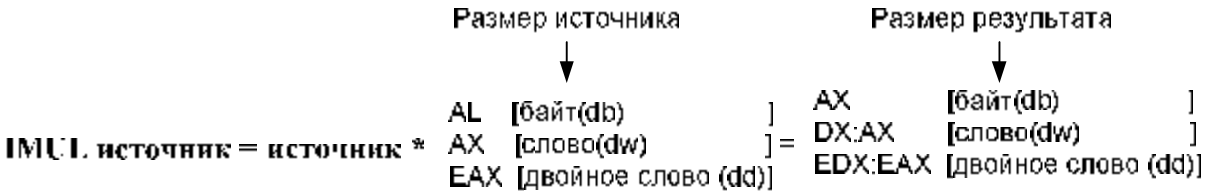
Состояние флагов после выполнения команды
(если ***старшая половина результата нулевая AH=0;DX=0;EDX=0***):

11	07	06	04	02	00
OF	SF	ZF	AF	PF	CF
1	?	?	?	?	0

Состояние флагов после выполнения команды:
(если ***старшая половина результата ненулевая AH≠0;DX≠0;EDX≠0***):

11	07	06	04	02	00
OF	SF	ZF	AF	PF	CF
1	?	?	?	?	1

IMUL источник



IMUL множ_1, множ_2 **(** IMUL AX, 5 AX = AX*5 **)**
IMUL рез-т, множ_1, множ_2 **(** IMUL DI, AX, 5 DI = AX*5 **)**

11	07	06	04	02	00
OF	SF	ZF	AF	PF	CF
r	?	?	?	?	r

2.2.4. Команды деления

Отметим, что деление выполняется как деление целых чисел, то есть, получим частное (целое) и остаток (целое). Следовательно, 19 : 5 = 3 и 4 - остаток. Никаких вещественных чисел не получим.

DIV (divide) - деления чисел без знака;

IDIV (integer divide) - деление целых чисел (со знаком).

DIV делитель

делимое / делитель	Размер делимого: Байт (db)	Слово (dw)	Двойное слово (dd)
делимое	AX	DX:AX	EDX:EAX
частное	AL	AX	EAX
остаток	AH	DX	EDX

Результаты команд деления на **флаги не влияет**.

Когда же частицу полностью нельзя разместить в отведенном ей слове или байте, то осуществляется прерывание типа 0 - деление на 0.

IDIV делитель

Остаток всегда имеет знак делимого. Знак частного зависит от состояния знаковых битов (старших разрядов) делимого и делителя.

2.2.5. Команды изменения знака

NEG (NEGate operand) - изменить знак операнда

NEG источник

Состояние флагов после выполнения команды (если результат нулевой):

11	07	06	04	02	00
OF	SF	ZF	AF	PF	CF
r	r	r	r	r	0

Состояние флагов после выполнения команды (если результат ненулевой):

11	07	06	04	02	00
OF	SF	ZF	AF	PF	CF
r	r	r	r	r	1

2.2.6. Команды расширения знака

Чтобы правильно подать код числа, которое записано в младшем байте, при считывании его из целого слова и код числа, записанного в младшее слово, при считывании его из младшего слова, нужно расширить знак числа соответственно на старший байт и старшее слово.

Для этого существуют две команды:

CBW (convert byte to word) - преобразование байта в слово;

CWD(convert word to double word) - преобразование слова в двойное слово;

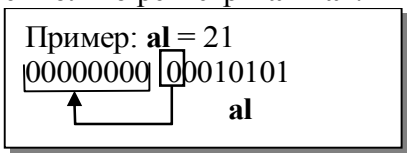
CWDE (convert word to double word Extended) - преобразование слова в двойное слово;

CDQ (Convert Double word to Quad word) - преобразование двойного слова в учетверенное слово.

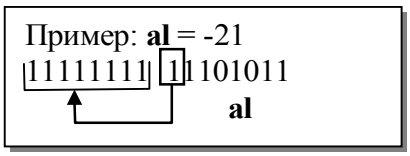
Алгоритм работы:

CBW — при работе команда использует только регистры **al** и **ax**:

анализ знакового бита регистра **al**:
если знаковый бит **al**=0, то **ah**=00h;



анализ знакового бита регистра **al**:
если знаковый бит **al**=1, то **ah**=0ffh



CWD — при работе команда использует только регистры **ax** и **dx**:
анализ знакового бита регистра al:
если знаковый бит **ax**=0, то **dx**=00h;
если знаковый бит **ax**=1, то **dx**=0ffh.

CWDE — при работе команда использует только регистры **ax** и **eax**:
анализ знакового бита регистра ax:
если знаковый бит **ax**=0, то установить старшее слово **eax**=0000h;
если знаковый бит **ax**=1, то установить старшее слово **eax**=0ffffh.
выполнение команды ***не влияет на флаги***

CDQ — при работе команда использует только регистры **eax** и **edx**:
анализ знакового бита регистра eax:
если знаковый бит **eax** =0, то установить старшее слово **edx** =0000h;
если знаковый бит **eax** =1, то установить старшее слово **edx** =0ffffh.
выполнение команды ***НЕ влияет на флаги***

Пример:

Данные команды используются для приведения операндов к нужной размерности с учетом знака. Такая необходимость может, в частности, возникнуть при программировании арифметических операций.

```
.386    ;только для cwde, cwd была для i8086
        mov     ebx,10fecd23h
        mov     ax,-3   ;ax=1111 1111 1111 1101
        cwde    ;eax=1111 1111 1111 1111 1111 1111 1111 1101
        add     eax,ebx
```