

Лекция 11. Логические команды, сдвиги, битовые команды

Логические команды

Выполняют **побитовые действия** над соответствующими битами двух операндов по правилам булевой алгебры:

- AND** - логическое И
- TEST** - логическое И без записи результата
- OR** - логическое ИЛИ
- XOR** - исключающее ИЛИ (сложение по модулю 2)
- NOT** - инверсия операнда

Все логические команды, кроме NOT, устанавливают арифметические флаги.

Булева алгебра

		Логическое И	Логическое ИЛИ	Исключаю- щее ИЛИ	Инверсия бита
a	b	$a \& b$	$a \vee b$	$a \oplus b$	\bar{a}
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Использование логического И

Логическое умножение AND и TEST используют для сброса в 0 отдельных битов первого операнда.

Подобранный для этих целей непосредственный второй операнд называют «маска». Для сброса отдельных битов в 0 маска должна иметь нули в этих разрядах, в остальных - 1

Пример: надо сбросить в 0 биты 2,3 и 4 в регистре bl.
Маска должна быть такой: 11100011

```
and bl, 11100011b ; биты bl:  ?????????? (произвольный код)
                    & 11100011 (маска)
```

```
                    -----
рез-т в bl:  ???000??
```

Использование логического ИЛИ

Логическое сложение OR используют для **установки в 1 отдельных битов первого операнда**. Второй непосредственный операнд является «маской».

Для установки отдельных битов в 1 маска должна иметь единицы в этих разрядах, в остальных – 0.

Пример: Установить в единицы биты 7-5 и 1-0 в регистре bl

```
or bl, 11100011b ; bl: ???????? (произвольный код)
                  v 11100011 (маска)
                  -----
                  bl: 111???11
```

Использование исключающего ИЛИ

Исключающее ИЛИ (сложение по модулю 2) XOR используют для **инвертирования отдельных битов первого операнда**. Второй непосредственный операнд - «маска». Для инвертирования отдельных битов маска должна иметь единицы в этих разрядах, в остальных – 0.

Пример: Инвертировать биты 7-5 и 1-0 в регистре bl

```
xor bl, 11100011b ; bl: аааааааа (произвольный код)
                   ⊕ 11100011 (маска)
                   -----
                   bl: āāāаааāā
```

Команды сдвигов

Выполняют сдвиг кодов, находящихся в регистре или памяти, на указанное количество бит

Количество бит задается 2-м операндом. Его можно указать двумя способами:

- однобайтным непосредственным числом (*i8*)
- предварительно занести в регистр *CL*.

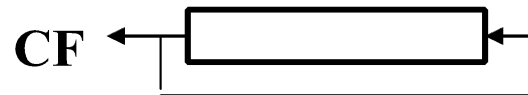
Общий синтаксис команд сдвига:

- Команда сдвига *r/m, i8* ; сдвиг на *i8* бит
- Команда сдвига *r/m, cl* ; сдвиг на количество бит, записанных в *cl*

Циклические сдвиги

□ ROL - циклический влево

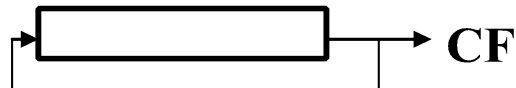
Выдвинутый бит заползает с стороны младших разрядов и дублируется в CF



Пример: циклический сдвиг однобайтного кода влево на 3 бита

bl= 01110000 → rol bl, 3 → bl= 10000011

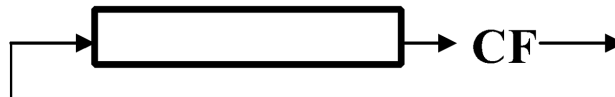
□ ROR –циклический вправо



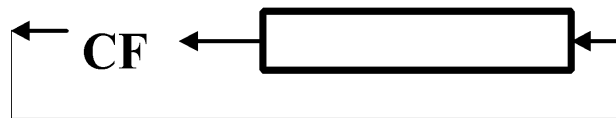
Циклические сдвиги через флаг CF

Флаг CF становится дополнительным битом в контуре сдвига

- **RCR** - циклический вправо через CF



- **RCL** - циклический влево через CF



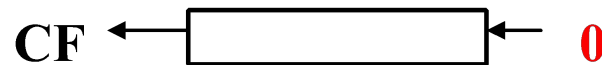
Пример: пусть, $bl = 11110001$ и флаг $cf = 0$

`rcr bl,1` ; в результате: $bl = 01111000$ и $cf = 1$

Простые сдвиги

В освобождающийся разряд заносится 0, а выдвигаемый бит попадает во флаг CF.

□ **SHL** – сдвиг влево



□ **SHR** – сдвиг вправо



Пример. Простой сдвиг однобайтного кода вправо на 3 бита
`bl = 11110001` → `shr bl, 3` → `bl = 00011110` и бит `cf = 0`

Выполнение умножения/деления простыми сдвигами

Простые сдвиги ВСЕГДА используют для умножения/ деления на величины, кратные степени двойки (2,4,8,16,32,64,128, ...)

- Сдвиг влево на 1 бит - умножение на 2
 - на 2 бита - на 4
 - на 3 бита = . . . на 8
 - на k бит = на 2^k
- Сдвиг вправо на 1 бит - деление на 2
 -
 - на k бит - деление на 2^k

Пример: Сдвиг влево на 3 = умножению на 8

bl= 0000011 (3_{10}) → `shl bl, 3` → bl=00011000 (24_{10})

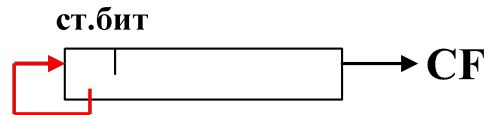
Арифметические сдвиги

Используются для сдвига **знаковых чисел**.

- **SAL** – арифметический сдвиг влево (аналогичен простому сдвигу SHL)



- **SAR** – арифметический сдвиг вправо. Освобождающиеся разряды заполняются значением знакового (старшего) разряда



Пример: деление на 4 знакового кода арифм.сдвигом вправо на 2
bl = 11111000 → sar bl, 2 → bl = 11111110
(-8₁₀) (-2₁₀)

Команды битовых операций

Поиск бита

- ❑ **BSF** reg, reg/mem ; Поиск самого **младшего** единичного бита в reg или mem. В первом reg возвращается **номер бита**.

Пример: пусть, bl= 0010**1**000

Результат выполнения команды **bsf al, bl** → al= 03h

- ❑ **BSR** reg, reg/mem ; Поиск самого **старшего** единичного бита в reg/ mem. В первом reg возвращается **номер бита**

Пример: пусть, bl= 00**1**01000

Результат выполнения команды **bsr al, bl** → al= 05h

Копирование бита в CF

- **BT** reg/mem,i8 ; из reg или mem во флаг CF копируется бит с номером i8

Пример:

bl=1100**1**011

Результат выполнения команды **bt bl, 3** → CF = **1**

Копирование бита в CF и преобразование исходного бита

- **BTC** reg/mem, i8 ; из reg/mem в CF копируется бит номер i8 и затем бит **инвертируется** в reg/mem
- **BTR** reg/mem, i8 ; из reg/mem в CF копируется бит номер i8 и затем он **сбрасывается** в reg/mem
- **BTS** reg /mem, i8 ; из reg/mem в CF копируется бит номер i8 и затем этот бит в reg/mem **устанавливается в 1**

Пример: пусть, bl= 1100**1**010

Результат выполнения команды **btr bl, 3** →

cf=**1** и bl= 1100**0**010