

# КОМАНДЫ ПЕРЕХОДА



## ФЛАГИ СОСТОЯНИЯ

- **Флаг переноса** (Carry flag или CF) устанавливается в случае, если при выполнении беззнаковой арифметической операции получается число, разрядность которого превышает разрядность выделенного для него поля результата.
- **Флаг переполнения** (Overflow flag, или OF) как и в предыдущем случае, но для операций со знаком.



## ФЛАГИ СОСТОЯНИЯ

- **Флаг знака** (Sign flag, или SF) устанавливается, если при выполнении арифметической или логической операции получается отрицательное число (т.е. старший бит результата равен 1).
- **Флаг нуля** (Zero flag, или ZF) устанавливается, если при выполнении арифметической или логической операции получается число, равное нулю (т.е. все биты результата равны 0).



## ФЛАГИ СОСТОЯНИЯ

- **Флаг служебного переноса** (Auxiliary Carry, или AF) устанавливается, если при выполнении арифметической операции с 8-разрядным операндом происходит перенос из третьего бита в четвертый.
- **Флаг чётности** (Parity flag, или PF) устанавливается в случае, если в результате выполнения арифметической или логической операции получается число, содержащее чётное количество битов.



# КОМАНДА CMP OP1,OP2

op1	op2	Числа со знаком	Числа без знака
>		ZF=0 и SF=OF	CF=0 и ZF=0
>=		SF=OF	CF=0
=		ZF=1	ZF=1
<=		ZF=1 и SF<>OF	CF=1 и ZF=1
<		SF<>OF (SF OF = 1)	CF=1



# КОМАНДЫ ПЕРЕХОДОВ

## Классификация переходов:

1. По модифицируемым регистрам.
  - NEAR – внутрисегментный, «ближний» (модифицируется только регистр IP);
  - FAR – межсегментный, «дальний» (модифицируются CS:IP)
2. По условию выполнения перехода.
  - безусловный – переход выполняется всегда;
  - условный – переход выполняется в случае, если комбинация проверяемых флагов истинна.
3. По способу задания адреса перехода.
  - Прямой – переход на заданную в программе метку.
  - Косвенный – переход по адресу, задаваемому через PОН.

Команда безусловного перехода:

**jmp адрес** – переход на метку/адрес

Пример.

jmp Label\_1 ; переход на инструкцию, помеченную меткой Label\_1

jmp [BX] ; переход на адрес, находящийся в памяти по адресу,  
; содержащемуся в BX



# КОМАНДЫ УСЛОВНЫХ ПЕРЕХОДОВ

Мнемокод	Аналог	Проверяемые флаги (условие перехода)	Используется для организации перехода, если...
jz	je	ZF=1	...результат=0 ...операнды равны
jnz	jne	ZF=0	...результат<>0 ...операнды не равны
js	-	SF=1	...результат отрицательный
jns	-	SF=0	...результат неотрицательный
jo	-	OF=1	... переполнение
jno	-	OF=0	...нет переполнения
jp	jpe	PF=1	... в результате четное число единиц
jn	jpo	PF=0	... в результате нечетное число единиц
jcxz	-	CX=0	... регистр CX (счетчик цикла) =0



# КОМАНДЫ УСЛОВНЫХ ПЕРЕХОДОВ ПРИ СРАВНЕНИИ БЕЗЗНАКОВЫХ ЧИСЕЛ

Мнемокод	Аналог	Проверяемые флаги (условие перехода)	Используется для организации перехода, если...
jb	jnae, jc	CF=1	... первый операнд «ниже» второго (при вычитании был перенос)
jnb	jae, jnc	CF=0	... первый операнд «выше» или равен второму
jbe	jna	CF or ZF = 1	... первый операнд «ниже» или равен второму
jnbe	ja	CF or ZF = 0	... первый операнд «выше» второго





# КОМАНДЫ УСЛОВНЫХ ПЕРЕХОДОВ ПРИ СРАВНЕНИИ ЗНАКОВЫХ ЧИСЕЛ

Мнемокод	Аналог	Проверяемые флаги (условие перехода)	Используется для организации перехода, если...
jl	jnge	$SF \oplus OF = 1$	... первый операнд меньше второго
jnl	jge	$SF \oplus OF = 0$	... первый операнд больше или равен второму
jle	jng	$(SF \oplus OF) \text{ or } ZF = 1$	... первый операнд меньше или равен второму
jnle	jg	$(SF \oplus OF) \text{ or } ZF = 1$	... первый операнд больше второго



# КОМАНДЫ ПЕРЕХОДОВ

Реализация аналогов условных операторов if и if-else языков высокого уровня в программе на ассемблере:

```
if (A>0) then  
    { Блок 1 }  
end if
```

**Вариант 1:**

```
    cmp AX, 0  
    jg Lab_1  
    jmp End_If  
Lab_1:  
    { Блок 1 }  
End_If: ...
```

**Вариант 2:**

```
    cmp AX, 0  
    jle End_If  
    { Блок 1 }  
End_If: ...
```

```
if (A>0) then  
    { Блок 1 }  
else  
    { Блок 2 }  
end if
```

**Вариант 1:**

```
    cmp AX, 0  
    jg Lab_1  
    { Блок 2 }  
    jmp End_If  
Lab_1:  
    { Блок 1 }  
End_If: ...
```

**Вариант 2:**

```
    cmp AX, 0  
    jle Lab_2  
    { Блок 1 }  
    jmp End_If  
Lab_2:  
    { Блок 2 }  
End_If: ...
```

# КОМАНДЫ ПЕРЕХОДОВ

Проверка нескольких условий в программе на ассемблере:

```
if (A>0) and (C=0) then  
  { Блок 1 }  
end if
```

```
cmp AX, 0  
jle End_If ; A<=0 – сразу выход  
cmp CX, 0  
jne End_If ; (A>0) and (C<>0) – выход  
{ Блок 1 }  
End_If: ...
```

```
if (A>0) or (C=0) then  
  { Блок 1 }  
end if
```

```
cmp AX, 0  
jg Lab_1 ; A>0 - Ок  
cmp CX, 0  
jne End_If ; (A<=0) and (C<>0) - выход  
Lab_1: { Блок 1 }  
End_If: ...
```