How `CMP` Affects CPU Flags in x86 Assembly

The `CMP` (Compare) instruction is essentially a **subtraction (`SUB`) without storing the result**. It updates the CPU flags based on the difference between two operands.

Syntax:

```
CMP destination, source
```

This performs:

```
destination - source
```

and updates the CPU flags accordingly.

Flags Affected by `CMP`

Flag	Description	How It's Affected by `CMP`
ZF (Zero Flag)	Set if the result is **zero** (`dest == src`)	`ZF = 1` if `dest == src` , otherwise `ZF = 0`
SF (Sign Flag)	Set if the result is **negative** (sign bit = 1)	`SF = 1` if result is negative (`MSB = 1`)
OF (Overflow Flag)	Set if signed overflow occurs	`OF = 1` if result is incorrect due to signed overflow
CF (Carry Flag)	Set if an unsigned borrow occurs	`CF = 1` if `dest < src` (unsigned borrow happened)
PF (Parity Flag)	Set if result has an even number of 1-bits	Updated based on result (rarely used)

Effects of `CMP` on Flags - Examples

1. When `CMP` Sets the Zero Flag (`ZF = 1`)

```
mov eax, 5
cmp eax, 5      ; 5 - 5 = 0 → ZF = 1 (equal)
je equal_label  ; Jumps because ZF = 1
```

2. When `CMP` Sets the Carry Flag (`CF = 1`) for Unsigned Comparisons

```
mov al, 5
cmp al, 10      ; 5 - 10 → CF = 1 (since 5 is smaller in unsigned)
jb below_label  ; Jumps because CF = 1 (unsigned comparison)
```

3. When `CMP` Sets the Sign Flag (`SF = 1`) for Signed Comparisons

```
mov al, -5
cmp al, 10      ; -5 - 10 = -15 → SF = 1 (negative result)
jl less_label   ; Jumps because SF = 1 (signed comparison)
```

4. When `CMP` Sets the Overflow Flag (`OF = 1`) for Signed Overflow

```
mov al, 127     ; Largest signed 8-bit value (+127)
cmp al, -128    ; 127 - (-128) → Overflow (out of range)
jo overflow_label ; Jumps if OF = 1
```

How Flags Are Used in Conditional Jumps

After `CMP` , conditional jumps determine whether the comparison was signed or unsigned:

Unsigned Jumps	Condition (Flags Used)
`JA` (Jump Above)	`CF = 0, ZF = 0` (`>` unsigned)
`JAE` (Jump Above or Equal)	`CF = 0` (`>=` unsigned)
`JB` (Jump Below)	`CF = 1` (`<` unsigned)
`JBE` (Jump Below or Equal)	`CF = 1 or ZF = 1` (`<=` unsigned)

Signed Jumps	Condition (Flags Used)
`JG` (Jump Greater)	`ZF = 0, SF = OF` (`>` signed)
`JGE` (Jump Greater or Equal)	`SF = OF` (`>=` signed)
`JL` (Jump Less)	`SF ≠ OF` (`<` signed)
`JLE` (Jump Less or Equal)	`ZF = 1 or SF ≠ OF` (`<=` signed)

Summary

- `CMP` performs `dest - src` and **does not store the result**, only updates flags.
- **Use `ZF` for equality checks** (`JE`, `JNE`).
- **Use `CF` for unsigned comparisons** (`JA`, `JB`).
- **Use `SF` and `OF` for signed comparisons** (`JG`, `JL`).
- **Signed overflow occurs when a result exceeds the signed number range** (`JO` for detecting it).

Would you like a specific example for a case you're working on? 🚀