Princeton University
COS 217: Introduction to Programming Systems
ARMv8 Condition Flags

Condition Flags

Bits in the `pstate` register

`CMP` *`Xs`*`|SP, `*`Xm`*

> CPU performs the subtraction *`Xs`*`|SP` − *`Xm`*
> More precisely, CPU performs the addition *`Xs`*`|SP + onescomp(`*`Xm`*`)` + 1 and sets the condition flags depending upon the sum:

Condition Code	
Z (zero flag)	CPU sets Z to 1 iff all bits of the sum are 0.
N (negative flag)	CPU sets N to 1 iff the most significant bit of the sum is 1.
C (carry flag)	CPU sets C to 1 iff the addition caused a carry.
V (overflow flag)	CPU sets V to 1 iff both addends are >= 0 and the sum is < 0, or both addends are < 0 and the sum is >= 0.

Conditional Branch Instructions (Used After Comparing Unsigned Numbers)

Instruction	Branch if and only if:
`beq (branch iff equal)`	`Z==1`
`bne (branch iff not equal)`	`Z==0`
`blo (branch iff lower)`	`C==0`
`bhs (branch iff higher or same)`	`C==1`
`bls (branch iff lower or same)`	`C==0 || Z==1`
`bhi (branch iff higher)`	`C==1 && Z==0`

Why does `blo` branch iff `C==0`? Examples (assuming a 4-bit computer):

(1) $5 - 3 = 0101_B - 0011_B = 0101_B + 1100_B + 1 = 0010_B$, C==1 => don't branch

(2) $5 - 0 = 0101_B - 0000_B = 0101_B + 1111_B + 1 = 1010_B$, C==1 => don't branch

(3) $3 - 5 = 0011_B - 0101_B = 0011_B + 1010_B + 1 = 1110_B$, C==0 => branch

(3) $0 - 5 = 0000_B - 0101_B = 0000_B + 1010_B + 1 = 1011_B$, C==0 => branch

So branch if and only if `C==0`.

Conditional Branch Instructions
(Used After Comparing Signed Numbers)

Instruction	Branch if and only if:
beq (branch iff equal)	Z==1
bne (branch iff not equal)	Z==0
blt (branch iff less than)	N!=V
bge (branch iff greater than or equal)	N==V
ble (branch iff less than or equal)	N!=V || Z==1
bgt (branch iff greater than)	N==V && Z==0

Why does `blt` branch iff if `N!=V`? Examples (assuming a 4 bit computer):

(1) 5 – 3 = 0101_B – 0011_B = 0101_B + 1100_B + 1 = 0010_B, N==0, V==0 => N==V => don't branch

(2) 3 – 5 = 0011_B – 0101_B = 0011_B + 1010_B + 1 = 1110_B, N==1, V==0 => N!=V => branch

(3) -5 – -3 = 1011_B – 1101_B = 1011_B + 0010_B + 1 = 1110_B, N==1, V==0 => N!=V => branch

(4) -3 – -5 = 1101_B – 1011_B = 1101_B + 0100_B + 1 = 0010_B, N==0, V==0 => N==V => don't branch

(5) 3 – -2 = 0011_B – 1110_B = 0011_B + 0001_B + 1 = 0101_B, N==0, V==0 => N==V => don't branch

(6) 3 – -6 = 0011_B – 1010_B = 0011_B + 0101_B + 1 = 1001_B, N==1, V==1 => N==V => don't branch

(7) -3 – 2 = 1101_B – 0010_B = 1101_B + 1101_B + 1 = 1111_B, N==1, V==0 => N!=V => branch

(8) -3 – 6 = 1101_B – 0110_B = 1101_B + 1001_B + 1 = 0111_B, N==0, V==1 => N!=V => branch

So branch if and only if `N!=V`.

Copyright © 2019 by Robert M. Dondero, Jr.