

WHILEHI (predicate as counter)

While decrementing unsigned scalar higher than scalar (predicate-as-counter)

Generate a predicate for a group of two or four vectors that starting from the highest numbered element of the group is true while the decrementing value of the first, unsigned scalar operand is higher than the second scalar operand and false thereafter down to the lowest numbered element of the group.

The full width of the scalar operands is significant for the purposes of comparison, and the full width first operand is decremented by one for each destination predicate element, irrespective of the predicate result element size.

The predicate result is placed in the predicate destination register using the predicate-as-counter encoding. Sets the first (N), none (Z), !last (C) condition flags based on the predicate result, and the V flag to zero.

SVE2

(FEAT_SVE2p1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	1	0	1	size		1	Rm				0	1	vl	0	1	0	Rn				1	1	PNd				
																		U			lt							eq			

WHILEHI <PNd>.<T>, <Xn>, <Xm>, <vl>

```
if !HaveSME2() && !HaveSVE2p1() then UNDEFINED;
constant integer esize = 8 << UInt(size);
constant integer rsize = 64;
integer n = UInt(Rn);
integer m = UInt(Rm);
integer d = UInt('1':PNd);
boolean unsigned = TRUE;
boolean invert = TRUE;
SVE_cmp op = Cmp_GT;
integer width = 2 << UInt(vl);
```

Assembler Symbols

<PNd> Is the name of the destination scalable predicate register PN8-PN15, with predicate-as-counter encoding, encoded in the "PNd" field.

<T>

Is the size specifier, encoded in “size”:

size	<T>
00	B
01	H
10	S
11	D

<Xn>

Is the 64-bit name of the first source general-purpose register, encoded in the “Rn” field.

<Xm>

Is the 64-bit name of the second source general-purpose register, encoded in the “Rm” field.

<vl>

Is the vl specifier, encoded in “vl”:

vl	<vl>
0	VLx2
1	VLx4

Operation

```
if HaveSVE2p1\(\) then CheckSVEEnabled\(\); else CheckStreamingSVEEnabled\(\);
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer elements = width * (VL DIV esize);
bits(rsize) operand1 = X[n, rsize];
bits(rsize) operand2 = X[m, rsize];
bits(PL) result;
boolean last = TRUE;
integer count = 0;

for e = elements-1 downto 0
    boolean cond;
    case op of
        when Cmp\_GT cond = (Int(operand1, unsigned) > Int(operand2, unsigned));
        when Cmp\_GE cond = (Int(operand1, unsigned) >= Int(operand2, unsigned));

    last = last && cond;
    if last then count = count + 1;
    operand1 = operand1 - 1;

result = EncodePredCount(esize, elements, count, invert, PL);
PSTATE.<N,Z,C,V> = PredCountTest(elements, count, invert);
P[d, PL] = result;
```

Operational information

If PSTATE.DIT is 1:

- The execution time of this instruction is independent of:
 - The values of the data supplied in any of its registers.
 - The values of the NZCV flags.

- The response of this instruction to asynchronous exceptions does not vary based on:
 - The values of the data supplied in any of its registers.
 - The values of the NZCV flags.

[Base
Instructions](#)

[SIMD&FP
Instructions](#)

[SVE
Instructions](#)

[SME
Instructions](#)

[Index by
Encoding](#)

[Sh
Pseudocode](#)

Internal version only: isa v33.64, AdvSIMD v29.12, pseudocode
no_diffs_2023_09_RC2, sve v2023-06_rel ; Build timestamp: 2023-09-18T17:56

Copyright Â© 2010-2023 Arm Limited or its affiliates. All rights reserved. This
document is Non-Confidential.