FMINNMQV

Floating-point minimum number recursive reduction of quadword vector segments

Floating-point minimum number of the same element numbers from each 128-bit source vector segment using a recursive pairwise reduction, placing each result into the corresponding element number of the 128-bit SIMD&FP destination register. Inactive elements in the source vector are treated as the default NaN.

Regardless of the value of FPCR.AH, the behavior is as follows:

- Negative zero compares less than positive zero.
- If one value is numeric and the other is a quiet NaN, the result is the numeric value.
- When FPCR.DN is 0, if either value is a signaling NaN or if both values are NaNs, the result is a quiet NaN.
- When FPCR.DN is 1, if either value is a signaling NaN or if both values are NaNs, the result is Default NaN.

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 1 1 0 0 1 0 0 size 0 1 0 1 0 1 0 1 Pg Zn Vd
```

```
FMINNMQV <Vd>.<T>, <Pq>, <Zn>.<Tb>
```

```
if !HaveSVE2p1() && !HaveSME2p1() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer n = UInt(Zn);
integer d = UInt(Vd);</pre>
```

Assembler Symbols

<Vd>

Is the name of the destination SIMD&FP register, encoded in the "Vd" field.

<T>

Is an arrangement specifier, encoded in "size":

size	<t></t>
00	RESERVED
01	8H
10	4S
11	2D

<Pg> Is the name of the governing scalable predicate register P0-P7, encoded in the "Pg" field.

Is the name of the source scalable vector register, encoded in the "Zn" field.

<Tb>

<Zn>

Is the size specifier, encoded in "size":

size	<tb></tb>
0.0	RESERVED
01	Н
10	S
11	D

Operation

```
CheckSVEEnabled();
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer segments = VL DIV 128;
constant integer elempersegment = 128 DIV esize;
bits(PL) mask = P[q, PL];
bits (VL) operand = if \underline{AnyActiveElement} (mask, esize) then \underline{Z}[n, VL] else
bits(esize) identity = FPDefaultNaN(esize);
bits (128) result = \frac{Zeros}{(128)};
constant integer p2bits = CeilPow2 (segments*esize);
constant integer p2elems = p2bits DIV esize;
for e = 0 to elempersegment-1
    bits(p2bits) stmp;
    bits(esize) dtmp;
    for s = 0 to p2elems-1
         if s < segments && <a href="ActivePredicateElement">ActivePredicateElement</a> (mask, s * elempersegments)
              Elem[stmp, s, esize] = Elem[operand, s * elempersegment + e
         else
              Elem[stmp, s, esize] = identity;
    dtmp = Reduce(ReduceOp_FMINNUM, stmp, esize);
    Elem[result, e, esize] = dtmp;
V[d, 128] = result;
```

BaseSIMD&FPSVESMEIndex byInstructionsInstructionsInstructionsInstructionsEncoding

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

Sh

Pseu

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