FMAXP

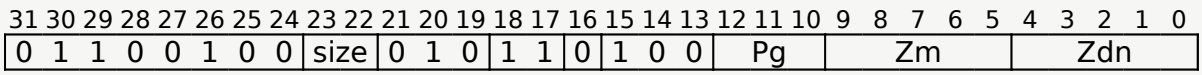
Floating-point maximum pairwise

Compute the maximum value of each pair of adjacent floating-point elements within each source vector, and interleave the results from corresponding lanes. The interleaved result values are destructively placed in the first source vector.
When FPCR.AH is 0, the behavior is as follows for each pairwise operation:

- Negative zero compares less than positive zero.
- When FPCR.DN is 0, if either element is a NaN, the result is a quiet NaN.
- When FPCR.DN is 1, if either element is a NaN, the result is Default NaN.

When FPCR.AH is 1, the behavior is as follows for each pairwise operation:

- If both elements are zeros, regardless of the sign of either zero, the result is the second element.
- If either element is a NaN, regardless of the value of FPCR.DN, the result is the second element.



FMAXP <Zdn>.<T>, <Pg>/M, <Zdn>.<T>, <Zm>.<T>

```
if !HaveSVE2() && !HaveSME() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer m = UInt(Zm);
integer dn = UInt(Zdn);
```

Assembler Symbols

<Zdn> Is the name of the first source and destination scalable vector register, encoded in the "Zdn" field.

<T> Is the size specifier, encoded in “size”:

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

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

Operation

```
CheckSVEEnabled();
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer elements = VL DIV esize;
bits(PL) mask = P[g, PL];
bits(VL) operand1 = Z[dn, VL];
bits(VL) operand2 = if AnyActiveElement(mask, esize) then Z[m, VL] else
bits(VL) result = Z[dn, VL];
bits(esize) element1;
bits(esize) element2;

for e = 0 to elements-1
    if ActivePredicateElement(mask, e, esize) then
        if IsEven(e) then
            element1 = Elem[operand1, e + 0, esize];
            element2 = Elem[operand1, e + 1, esize];
        else
            element1 = Elem[operand2, e - 1, esize];
            element2 = Elem[operand2, e + 0, esize];
        Elem[result, e, esize] = FPMax(element1, element2, FPCR[]);

Z[dn, VL] = result;
```

Operational information

This instruction might be immediately preceded in program order by a MOVPRFX instruction. The MOVPRFX instruction must conform to all of the following requirements, otherwise the behavior of the MOVPRFX and this instruction is unpredictable:

- The MOVPRFX instruction must be unpredicated.
- The MOVPRFX instruction must specify the same destination register as this instruction.
- The destination register must not refer to architectural register state referenced by any other source operand register of this instruction.

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