BFCLAMP

Multi-vector BFloat16 floating-point clamp to minimum/maximum number

Clamp each BFloat16 element in the two or four destination vectors to between the BFloat16 minimum value in the corresponding element of the first source vector and the BFloat16 maximum value in the corresponding element of the second source vector and destructively place the clamped results in the corresponding elements of the two or four destination vectors.

Regardless of the value of FPCR.AH, the behavior is as follows for each minimum number and maximum number operation:

- 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.

This instruction follows SME2.1 non-widening BFloat16 numerical behaviors corresponding to instructions that place their results in two or four SVE Z vectors.

This instruction is unpredicated.

ID_AA64SMFR0_EL1.B16B16 indicates whether this instruction is implemented.

It has encodings from 2 classes: Two registers and Four registers

Two registers

(FEAT_SVE_B16B16)

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
1 1 0 0 0 0 0 1	0	0	1	Zm	1 1 0 0 0 0	Zn	Zd	0
	size<1>	size<0>						

BFCLAMP { <Zd1>.H-<Zd2>.H }, <Zn>.H, <Zm>.H

```
if !HaveSME2() || !IsFeatureImplemented(FEAT_SVE_B16B16) then UNDEFINED
integer n = UInt(Zn);
integer m = UInt(Zm);
integer d = UInt(Zd:'0');
constant integer nreg = 2;
```

Four registers

(FEAT_SVE_B16B16)

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
1 1 0 0 0 0 0 1	0	0	1	Zm	1 1 0 0 1 0	Zn	Zd	0	0
	size<1>	size<0>							

```
    BFCLAMP { <Zd1>.H-<Zd4>.H }, <Zn>.H, <Zm>.H

if !HaveSME2() || !IsFeatureImplemented(FEAT_SVE_B16B16) then UNDEFINED
integer n = UInt(Zn);
integer m = UInt(Zm);
integer d = UInt(Zd:'00');
constant integer nreg = 4;
```

Assembler Symbols

<Zd1>	For the two registers variant: is the name of the first destination scalable vector register of a multi-vector sequence, encoded as "Zd" times 2.
	For the four registers variant: is the name of the first destination scalable vector register of a multi-vector sequence, encoded as "Zd" times 4.
<Zd4>	Is the name of the fourth destination scalable vector register of a multi-vector sequence, encoded as "Zd" times 4 plus 3.
<Zd2>	Is the name of the second destination scalable vector register of a multi-vector sequence, encoded as "Zd" times 2 plus 1.
<Zn>	Is the name of the first source scalable vector register, encoded in the "Zn" field.
<Zm>	Is the name of the second source scalable vector register, encoded in the "Zm" field.

Operation

```
CheckStreamingSVEEnabled();
constant integer VL = CurrentVL;
constant integer elements = VL DIV 16;
array [0..3] of bits(VL) results;

for r = 0 to nreg-1
    bits(VL) operand1 = Z[n, VL];
    bits(VL) operand2 = Z[m, VL];
    bits(VL) operand3 = Z[d+r, VL];
    for e = 0 to elements-1
        bits(16) element1 = Elem[operand1, e, 16];
        bits(16) element2 = Elem[operand2, e, 16];
        bits(16) element3 = Elem[operand3, e, 16];
        Elem[results[r], e, 16] = BFMinNum(BFMaxNum(element1, element3,

for r = 0 to nreg-1
    Z[d+r, VL] = results[r];
```
