FCM<cc> (zero)

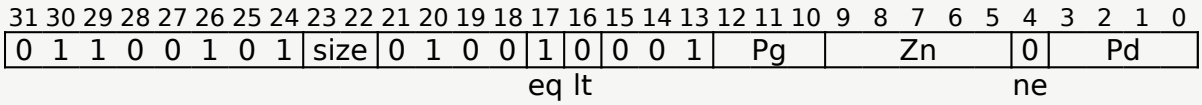
Floating-point compare vector with zero

Compare active floating-point elements in the source vector with zero, and place the boolean results of the specified comparison in the corresponding elements of the destination predicate. Inactive elements in the destination predicate register are set to zero. Does not set the condition flags.

<cc>	Comparison
EQ	equal
GE	greater than or equal
GT	greater than
LE	less than or equal
LT	less than
NE	not equal
UO	unordered

It has encodings from 6 classes: Equal , Greater than , Greater than or equal , Less than , Less than or equal and Not equal

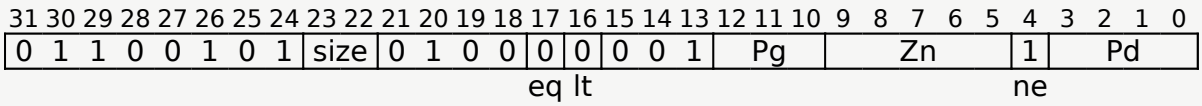
Equal



FCMEQ <Pd>.<T>, <Pg>/Z, <Zn>.<T>, #0.0

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer n = UInt(Zn);
integer d = UInt(Pd);
SVECmp op = Cmp_EQ;
```

Greater than

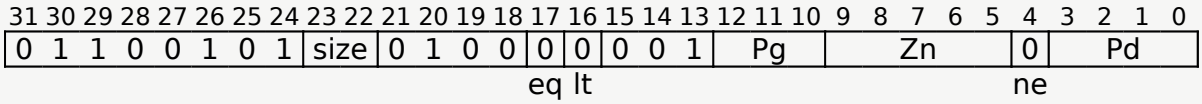


FCMGT <Pd>.<T>, <Pg>/Z, <Zn>.<T>, #0.0

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer n = UInt(Zn);
```

```
integer d = UInt(Pd);
SVECmp op = Cmp_GT;
```

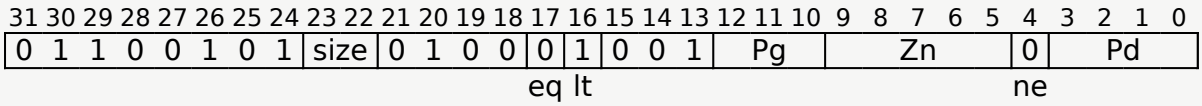
Greater than or equal



FCMGE <Pd>.<T>, <Pg>/Z, <Zn>.<T>, #0.0

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer n = UInt(Zn);
integer d = UInt(Pd);
SVECmp op = Cmp_GE;
```

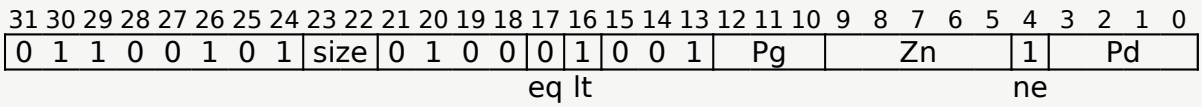
Less than



FCMLT <Pd>.<T>, <Pg>/Z, <Zn>.<T>, #0.0

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer n = UInt(Zn);
integer d = UInt(Pd);
SVECmp op = Cmp_LT;
```

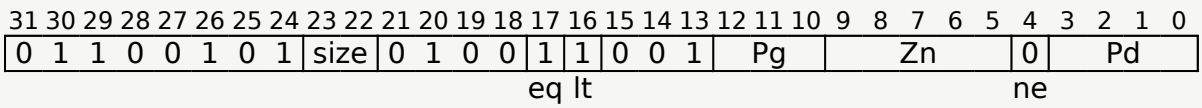
Less than or equal



FCMLE <Pd>.<T>, <Pg>/Z, <Zn>.<T>, #0.0

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer n = UInt(Zn);
integer d = UInt(Pd);
SVECmp op = Cmp_LE;
```

Not equal



FCMNE <Pd>.<T>, <Pg>/Z, <Zn>.<T>, #0.0

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer n = UInt(Zn);
integer d = UInt(Pd);
SVECmp op = Cmp_NE;
```

Assembler Symbols

<Pd> Is the name of the destination scalable predicate register, encoded in the "Pd" 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.

<Zn> Is the name of the source scalable vector register, encoded in the "Zn" 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) operand = if AnyActiveElement(mask, esize) then Z[n, VL] else
bits(PL) result;
constant integer psize = esize DIV 8;

for e = 0 to elements-1
    if ActivePredicateElement(mask, e, esize) then
        bits(esize) element = Elem[operand, e, esize];
        boolean res;
        case op of
            when Cmp_EQ res = FPCompareEQ(element, 0<esize-1:0>, FPCR[]
            when Cmp_GE res = FPCompareGE(element, 0<esize-1:0>, FPCR[]
            when Cmp_GT res = FPCompareGT(element, 0<esize-1:0>, FPCR[]
            when Cmp_NE res = FPCompareNE(element, 0<esize-1:0>, FPCR[]
            when Cmp_LT res = FPCompareGT(0<esize-1:0>, element, FPCR[]
            when Cmp_LE res = FPCompareGE(0<esize-1:0>, element, FPCR[]
        bit pbit = if res then '1' else '0';
        Elem[result, e, psize] = ZeroExtend(pbit, psize);
    else
        Elem[result, e, psize] = ZeroExtend('0', psize);
```

```
    P[d, PL] = result;
```

Operational information

If FEAT_SME is implemented and the PE is in Streaming SVE mode, then any subsequent instruction which is dependent on the predicate register written by this instruction might be significantly delayed.

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