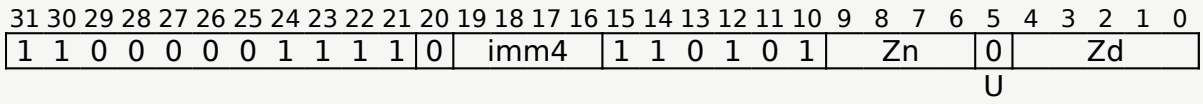
SQRSHR (two registers)

Multi-vector signed saturating rounding shift right narrow by immediate

Shift right by an immediate value, the signed integer value in each element of the two source vectors and place the rounded results in the half-width destination elements. Each result element is saturated to the half-width N-bit element's signed integer range $-2^{(N-1)}$ to $(2^{(N-1)})-1$. The immediate shift amount is an unsigned value in the range 1 to 16.
This instruction is unpredicated.

SME2
(FEAT_SME2)




```
SQRSHR <Zd>.H, { <Zn1>.S-<Zn2>.S }, #<const>
```

```
if !HaveSME2() then UNDEFINED;
constant integer esize = 16;
integer n = UInt(Zn:'0');
integer d = UInt(Zd);
integer shift = esize - UInt(imm4);
```

Assembler Symbols

<Zd>	Is the name of the destination scalable vector register, encoded in the "Zd" field.
<Zn1>	Is the name of the first scalable vector register of a multi-vector sequence, encoded as "Zn" times 2.
<Zn2>	Is the name of the second scalable vector register of a multi-vector sequence, encoded as "Zn" times 2 plus 1.
<const>	Is the immediate shift amount, in the range 1 to 16, encoded in the "imm4" field.

Operation

```
CheckStreamingSVEEnabled();
constant integer VL = CurrentVL;
constant integer elements = VL DIV (2 * esize);
bits(VL) result;

for r = 0 to 1
    bits(VL) operand = Z[n+r, VL];
    for e = 0 to elements-1
        bits(2 * esize) element = Elem[operand, e, 2 * esize];
```

```
            integer res = (SInt(element) + (1 << (shift-1))) >> shift;
            Elem[result, r*elements + e, esize] = SignedSat(res, esize);

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

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