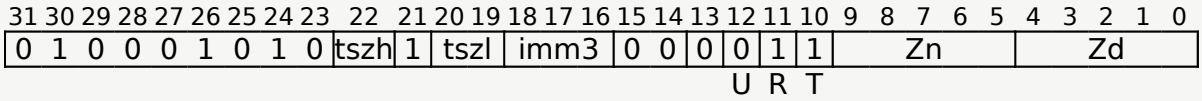
SQRSHRUNT

Signed saturating rounding shift right unsigned narrow by immediate (top)

Shift each signed integer value in the source vector elements right by an immediate value, and place the rounded results in the odd-numbered half-width destination elements, leaving the even-numbered elements unchanged. Each result element is saturated to the half-width N-bit element's unsigned integer range 0 to (2^N)-1. The immediate shift amount is an unsigned value in the range 1 to number of bits per element. This instruction is unpredicated.




```
SQRSHRUNT <Zd>.<T>, <Zn>.<Tb>, #<const>
```

```
if !HaveSVE2() && !HaveSME() then UNDEFINED;
bits(3) tsize = tszh:tszl;
if tsize == '000' then UNDEFINED;
constant integer esize = 8 << HighestSetBit(tsize);
integer n = UInt(Zn);
integer d = UInt(Zd);
integer shift = (2 * esize) - UInt(tsize:imm3);
```

Assembler Symbols

<Zd> Is the name of the destination scalable vector register, encoded in the "Zd" field.

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

tszh	tszl	<T>
0	00	RESERVED
0	01	B
0	1x	H
1	xx	S

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

<Tb>

Is the size specifier, encoded in “tszh:tszl”:

tszh	tszl	<Tb>
0	00	RESERVED
0	01	H
0	1x	S
1	xx	D

<const>

Is the immediate shift amount, in the range 1 to number of bits per element, encoded in "tszh:tszl:imm3".

Operation

```
CheckSVEEnabled();
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer elements = VL DIV (2 * esize);
bits(VL) operand = Z[n, VL];
bits(VL) result = Z[d, 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, 2*e + 1, esize] = UnsignedSat(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