SMULLT (indexed)

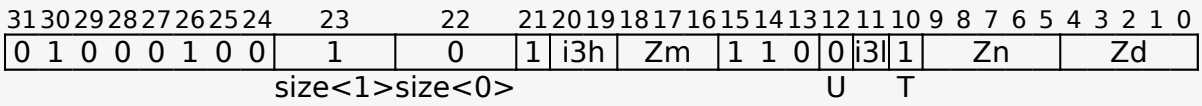
Signed multiply long (top, indexed)

Multiply the odd-numbered signed elements within each 128-bit segment of the first source vector by the specified element in the corresponding second source vector segment, and place the results in the overlapping double-width elements of the destination vector register.

The elements within the second source vector are specified using an immediate index which selects the same element position within each 128-bit vector segment. The index range is from 0 to one less than the number of elements per 128-bit segment, encoded in 2 or 3 bits depending on the size of the element.

It has encodings from 2 classes: 32-bit and 64-bit

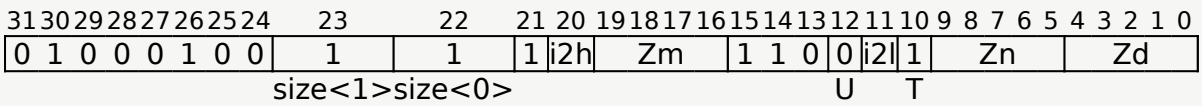
32-bit



SMULLT <Zd>.S, <Zn>.H, <Zm>.H[<imm>]

```
if !HaveSVE2() && !HaveSME() then UNDEFINED;
constant integer esize = 16;
integer index = UInt(i3h:i3l);
integer n = UInt(Zn);
integer m = UInt(Zm);
integer d = UInt(Zd);
integer sel = 1;
```

64-bit



SMULLT <Zd>.D, <Zn>.S, <Zm>.S[<imm>]

```
if !HaveSVE2() && !HaveSME() then UNDEFINED;
constant integer esize = 32;
integer index = UInt(i2h:i2l);
integer n = UInt(Zn);
integer m = UInt(Zm);
integer d = UInt(Zd);
integer sel = 1;
```

Assembler Symbols

<Zd>	Is the name of the destination scalable vector register, encoded in the "Zd" field.
<Zn>	Is the name of the first source scalable vector register, encoded in the "Zn" field.
<Zm>	For the 32-bit variant: is the name of the second source scalable vector register Z0-Z7, encoded in the "Zm" field. For the 64-bit variant: is the name of the second source scalable vector register Z0-Z15, encoded in the "Zm" field.
<imm>	For the 32-bit variant: is the element index, in the range 0 to 7, encoded in the "i3h:i3l" fields. For the 64-bit variant: is the element index, in the range 0 to 3, encoded in the "i2h:i2l" fields.

Operation

```
CheckSVEEnabled();
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer elements = VL DIV (2 * esize);
constant integer eltspersegment = 128 DIV (2 * esize);
bits(VL) operand1 = Z[n, VL];
bits(VL) operand2 = Z[m, VL];
bits(VL) result;

for e = 0 to elements-1
    integer s = e - (e MOD eltspersegment);
    integer element1 = SInt(Elem[operand1, 2 * e + sel,   esize]);
    integer element2 = SInt(Elem[operand2, 2 * s + index, esize]);
    integer res = element1 * element2;
    Elem[result, e, 2*esize] = res<2*esize-1:0>;

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

Operational information

If FEAT_SVE2 is implemented or FEAT_SME is implemented, then if PSTATE.DIT is 1:

- The execution time of this instruction is independent of:
  - The values of the data supplied in any of its registers.
  - The values of the NZCV flags.
- The response of this instruction to asynchronous exceptions does not vary based on:
  - The values of the data supplied in any of its registers.
  - The values of the NZCV flags.

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