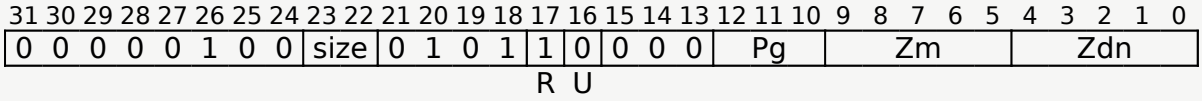
SDIVR

Signed reversed divide (predicated)

Signed reversed divide active elements of the second source vector by corresponding elements of the first source vector and destructively place the quotient in the corresponding elements of the first source vector. Inactive elements in the destination vector register remain unmodified.



SDIVR <Zdn>.<T>, <Pg>/M, <Zdn>.<T>, <Zm>.<T>

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
if size IN {'0x'} then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer dn = UInt(Zdn);
integer m = UInt(Zm);
boolean unsigned = FALSE;
```

Assembler Symbols

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

<T> Is the size specifier, encoded in “size<0>”:

size<0>	<T>
0	S
1	D

<Pg> Is the name of the governing scalable predicate register P0-P7, encoded in the "Pg" field.

<Zm> Is the name of the second source scalable vector register, encoded in the "Zm" 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) operand1 = Z[dn, VL];
bits(VL) operand2 = if AnyActiveElement(mask, esize) then Z[m, VL] else
bits(VL) result;
```

```
for e = 0 to elements-1
    integer element1 = Int(Elem[operand1, e, esize], unsigned);
    if ActivePredicateElement(mask, e, esize) then
        integer element2 = Int(Elem[operand2, e, esize], unsigned);
        integer quotient;
        if element1 == 0 then
            quotient = 0;
        else
            quotient = RoundTowardsZero(Real(element2) / Real(element1));
        Elem[result, e, esize] = quotient<esize-1:0>;
    else
        Elem[result, e, esize] = Elem[operand1, e, esize];

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

Operational information

This instruction might be immediately preceded in program order by a MOVPRFX instruction. The MOVPRFX instruction must conform to all of the following requirements, otherwise the behavior of the MOVPRFX and this instruction is unpredictable:

- The MOVPRFX instruction must be unpredicated, or be predicated using the same governing predicate register and source element size as this instruction.
- The MOVPRFX instruction must specify the same destination register as this instruction.
- The destination register must not refer to architectural register state referenced by any other source operand register of this instruction.

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