LDNT1SH

Gather load non-temporal signed halfwords

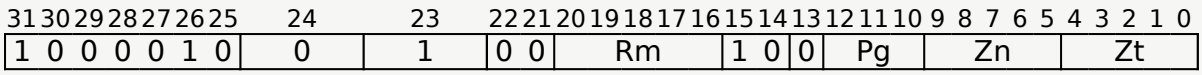
Gather load non-temporal of signed halfwords to active elements of a vector register from memory addresses generated by a vector base plus a 64-bit unscaled scalar register offset. Inactive elements will not cause a read from Device memory or signal faults, and are set to zero in the destination vector.

A non-temporal load is a hint to the system that this data is unlikely to be referenced again soon.

This instruction is illegal when executed in Streaming SVE mode, unless FEAT_SME_FA64 is implemented and enabled.

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

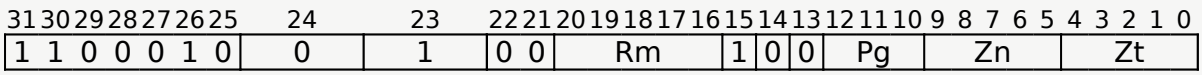
32-bit unscaled offset



LDNT1SH { <Zt>.S }, <Pg>/Z, [<Zn>.S{, <Xm>}]

```
if !HaveSVE2() then UNDEFINED;
integer t = UInt(Zt);
integer n = UInt(Zn);
integer m = UInt(Rm);
integer g = UInt(Pg);
constant integer esize = 32;
constant integer msize = 16;
boolean unsigned = FALSE;
```

64-bit unscaled offset



LDNT1SH { <Zt>.D }, <Pg>/Z, [<Zn>.D{, <Xm>}]

```
if !HaveSVE2() then UNDEFINED;
integer t = UInt(Zt);
integer n = UInt(Zn);
integer m = UInt(Rm);
integer g = UInt(Pg);
constant integer esize = 64;
constant integer msize = 16;
boolean unsigned = FALSE;
```

Assembler Symbols

<Zt>	Is the name of the scalable vector register to be transferred, encoded in the "Zt" field.
<Pg>	Is the name of the governing scalable predicate register P0-P7, encoded in the "Pg" field.
<Zn>	Is the name of the base scalable vector register, encoded in the "Zn" field.
<Xm>	Is the optional 64-bit name of the general-purpose offset register, defaulting to XZR, encoded in the "Rm" field.

Operation

```
CheckNonStreamingSVEEnabled();
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer elements = VL DIV esize;
bits(PL) mask = P[g, PL];
bits(VL) base;
bits(64) offset;
bits(VL) result;
bits(msize) data;
constant integer mbytes = msize DIV 8;
boolean contiguous = FALSE;
boolean nontemporal = TRUE;
boolean tagchecked = TRUE;
AccessDescriptor accdesc = CreateAccDescSVE(MemOp_LOAD, nontemporal, co

if AnyActiveElement(mask, esize) then
    base = Z[n, VL];
    offset = X[m, 64];

for e = 0 to elements-1
    if ActivePredicateElement(mask, e, esize) then
        bits(64) addr = ZeroExtend(Elem[base, e, esize], 64) + offset;
        data = Mem[addr, mbytes, accdesc];
        Elem[result, e, esize] = Extend(data, esize, unsigned);
    else
        Elem[result, e, esize] = Zeros(esize);

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

Operational information

If FEAT_SVE2 is implemented or FEAT_SME is implemented, then if PSTATE.DIT is 1, the timing of this instruction is insensitive to the value of the data being loaded or stored when its governing predicate register contains the same value for each execution.

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