LD1H (vector plus immediate)

Gather load unsigned halfwords to vector (immediate index)

Gather load of unsigned halfwords to active elements of a vector register from memory addresses generated by a vector base plus immediate index. The index is a multiple of 2 in the range 0 to 62. Inactive elements will not cause a read from Device memory or signal faults, and are set to zero in the destination vector.

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 element](#) and [64-bit element](#)

32-bit element

31 30 29 28 27 26 25	24	23	22 21	20 19 18 17 16	15	14	13	12 11 10	9 8 7 6 5	4 3 2 1 0
1 0 0 0 0 1 0	0	1	0 1	imm5	1	1	0	Pg	Zn	Zt
	msz<1> msz<0>				U	ff				

LD1H { <Zt>.S }, <Pg>/Z, [<Zn>.S{, #<imm>}]

```
if !HaveSVE() then UNDEFINED;
integer t = UInt(Zt);
integer n = UInt(Zn);
integer g = UInt(Pg);
constant integer esize = 32;
constant integer msize = 16;
boolean unsigned = TRUE;
integer offset = UInt(imm5);
```

64-bit element

31 30 29 28 27 26 25	24	23	22 21	20 19 18 17 16	15	14	13	12 11 10	9 8 7 6 5	4 3 2 1 0
1 1 0 0 0 1 0	0	1	0 1	imm5	1	1	0	Pg	Zn	Zt
	msz<1> msz<0>				U	ff				

LD1H { <Zt>.D }, <Pg>/Z, [<Zn>.D{, #<imm>}]

```
if !HaveSVE() then UNDEFINED;
integer t = UInt(Zt);
integer n = UInt(Zn);
integer g = UInt(Pg);
constant integer esize = 64;
constant integer msize = 16;
boolean unsigned = TRUE;
integer offset = UInt(imm5);
```

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.

<imm>   Is the optional unsigned immediate byte offset, a multiple of
                2 in the range 0 to 62, defaulting to 0, encoded in the
                "imm5" 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(VL) result;
bits(msize) data;
constant integer mbytes = msize DIV 8;
boolean contiguous = FALSE;
boolean nontemporal = FALSE;
boolean tagchecked = TRUE;
AccessDescriptor accdesc = CreateAccDescSVE(MemOp_LOAD, nontemporal, co

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

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.