PRFW (scalar plus vector)

Gather prefetch words (scalar plus vector)

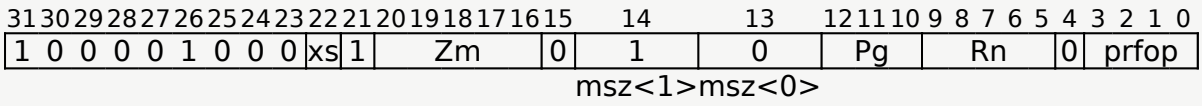
Gather prefetch of words from the active memory addresses generated by a 64-bit scalar base plus vector index. The index values are optionally first sign or zero-extended from 32 to 64 bits and then multiplied by 4. Inactive addresses are not prefetched from memory.

The <prfop> symbol specifies the prefetch hint as a combination of three options: access type PLD for load or PST for store; target cache level L1, L2 or L3; temporality (KEEP for temporal or STRM for non-temporal).

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

It has encodings from 3 classes: 32-bit scaled offset , 32-bit unpacked scaled offset and 64-bit scaled offset

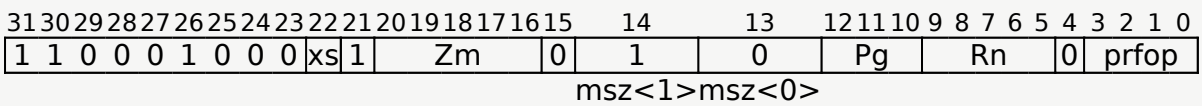
32-bit scaled offset



PRFW <prfop>, <Pg>, [<Xn|SP>, <Zm>.S, <mod> #2]

```
if !HaveSVE() then UNDEFINED;
constant integer esize = 32;
integer g = UInt(Pg);
integer n = UInt(Rn);
integer m = UInt(Zm);
integer level = UInt(prfop<2:1>);
boolean stream = (prfop<0> == '1');
pref_hint = if prfop<3> == '0' then Prefetch_READ else Prefetch_WRITE;
constant integer offs_size = 32;
boolean offs_unsigned = (xs == '0');
integer scale = 2;
```

32-bit unpacked scaled offset

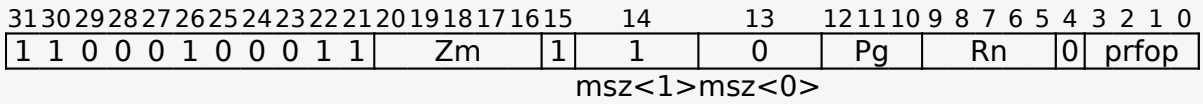


PRFW <prfop>, <Pg>, [<Xn|SP>, <Zm>.D, <mod> #2]

```
if !HaveSVE() then UNDEFINED;
constant integer esize = 64;
integer g = UInt(Pg);
integer n = UInt(Rn);
integer m = UInt(Zm);
integer level = UInt(prfop<2:1>);
boolean stream = (prfop<0> == '1');
```

```
pref_hint = if prfop<3> == '0' then Prefetch_READ else Prefetch_WRITE;
constant integer offs_size = 32;
boolean offs_unsigned = (xs == '0');
integer scale = 2;
```

64-bit scaled offset



PRFW <prfop>, <Pg>, [<Xn|SP>, <Zm>.D, LSL #2]

```
if !HaveSVE() then UNDEFINED;
constant integer esize = 64;
integer g = UInt(Pg);
integer n = UInt(Rn);
integer m = UInt(Zm);
integer level = UInt(prfop<2:1>);
boolean stream = (prfop<0> == '1');
pref_hint = if prfop<3> == '0' then Prefetch_READ else Prefetch_WRITE;
constant integer offs_size = 64;
boolean offs_unsigned = TRUE;
integer scale = 2;
```

Assembler Symbols

<prfop> Is the prefetch operation specifier, encoded in “prfop”:

prfop	<prfop>
0000	PLDL1KEEP
0001	PLDL1STRM
0010	PLDL2KEEP
0011	PLDL2STRM
0100	PLDL3KEEP
0101	PLDL3STRM
x11x	#uimm4
1000	PSTL1KEEP
1001	PSTL1STRM
1010	PSTL2KEEP
1011	PSTL2STRM
1100	PSTL3KEEP
1101	PSTL3STRM

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

<Xn|SP> Is the 64-bit name of the general-purpose base register or stack pointer, encoded in the "Rn" field.

<Zm> Is the name of the offset scalable vector register, encoded in the "Zm" field.

<mod>

Is the index extend and shift specifier, encoded in “xs”:

xs	<mod>
0	UXTW
1	SXTW

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(64) base;
bits(VL) offset;

if AnyActiveElement(mask, esize) then
    base = if n == 31 then SP[] else X[n, 64];
    offset = Z[m, VL];

for e = 0 to elements-1
    if ActivePredicateElement(mask, e, esize) then
        integer off = Int(Elem[offset, e, esize]<offs_size-1:0>, offs_unsig
        bits(64) addr = base + (off << scale);
        Hint_Prefetch(addr, pref_hint, level, stream);
```


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