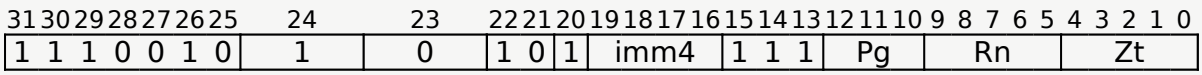
ST3W (scalar plus immediate)

Contiguous store three-word structures from three vectors (immediate index)

Contiguous store three-word structures, each from the same element number in three vector registers to the memory address generated by a 64-bit scalar base and an immediate index which is a multiple of 3 in the range -24 to 21 that is multiplied by the vector's in-memory size, irrespective of predication,

Each predicate element applies to the same element number in each of the three vector registers, or equivalently to the three consecutive words in memory which make up each structure. Inactive structures are not written to memory.




```
ST3W { <Zt1>.S, <Zt2>.S, <Zt3>.S }, <Pg>, [<Xn|SP>{, #<imm>, MUL VL}]
```

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
integer t = UInt(Zt);
integer n = UInt(Rn);
integer g = UInt(Pg);
constant integer esize = 32;
integer offset = SInt(imm4);
constant integer nreg = 3;
```

Assembler Symbols

<Zt1>	Is the name of the first scalable vector register to be transferred, encoded in the "Zt" field.
<Zt2>	Is the name of the second scalable vector register to be transferred, encoded as "Zt" plus 1 modulo 32.
<Zt3>	Is the name of the third scalable vector register to be transferred, encoded as "Zt" plus 2 modulo 32.
<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.
<imm>	Is the optional signed immediate vector offset, a multiple of 3 in the range -24 to 21, defaulting to 0, encoded in the "imm4" field.

Operation

```
CheckSVEEnabled();
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer elements = VL DIV esize;
bits(64) base;
bits(PL) mask = P[g, PL];
constant integer mbytes = esize DIV 8;
array [0..2] of bits(VL) values;
boolean contiguous = TRUE;
boolean nontemporal = FALSE;
boolean tagchecked = n != 31;
AccessDescriptor accdesc = CreateAccDescSVE(MemOp_STORE, nontemporal, c

if !AnyActiveElement(mask, esize) then
    if n == 31 && ConstrainUnpredictableBool(Unpredictable_CHECKSPNONEA
        CheckSPAlignment();
else
    if n == 31 then CheckSPAlignment();
    base = if n == 31 then SP[] else X[n, 64];

for r = 0 to nreg-1
    values[r] = Z[(t+r) MOD 32, VL];

for e = 0 to elements-1
    for r = 0 to nreg-1
        if ActivePredicateElement(mask, e, esize) then
            integer eoff = (offset * elements * nreg) + (e * nreg) + r;
            bits(64) addr = base + eoff * mbytes;
            Mem[addr, mbytes, accdesc] = Elem[values[r], e, esize];
```

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