STR (vector)

Store ZA array vector

The ZA array vector is selected by the sum of the vector select register and immediate offset, modulo the number of bytes in a Streaming SVE vector. The immediate offset is in the range 0 to 15. The memory address is generated by a 64-bit scalar base, plus the same optional immediate offset multiplied by the current vector length in bytes. This instruction is unpredicated.

The store is performed as contiguous byte accesses, with no endian conversion and no guarantee of single-copy atomicity larger than a byte. However, if alignment is checked, then the base register must be aligned to 16 bytes.

This instruction does not require the PE to be in Streaming SVE mode, and it is expected that this instruction will not experience a significant slowdown due to contention with other PEs that are executing in Streaming SVE mode.

SME
(FEAT_SME)

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	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	Rv		0	0	0	Rn					0	off4			

STR ZA[<Wv>, <offs>], [<Xn|SP>{, #<offs>, MUL VL}]

```
if !HaveSME() then UNDEFINED;
integer n = UInt(Rn);
integer v = UInt('011':Rv);
integer offset = UInt(off4);
```

Assembler Symbols

<Wv>	Is the 32-bit name of the vector select register W12-W15, encoded in the "Rv" field.
<offs>	Is the vector select offset and optional memory offset, in the range 0 to 15, defaulting to 0, encoded in the "off4" field.
<Xn|SP>	Is the 64-bit name of the general-purpose base register or stack pointer, encoded in the "Rn" field.

Operation

```
CheckSMEAndZAEnabled();
constant  integer SVL = CurrentSVL;
constant  integer dim = SVL DIV 8;
bits(64)  base;
integer   moffs = offset * dim;
bits(SVL) src;
```

```
bits(32)  vbase = X[v, 32];
integer   vec = (UInt(vbase) + offset) MOD dim;
boolean   contiguous = TRUE;
boolean   nontemporal = FALSE;
boolean   tagchecked = n != 31;
AccessDescriptor accdesc = CreateAccDescSME(MemOp_STORE, nontemporal, c

if HaveTME() && TSTATE.depth > 0 then
    FailTransaction(TMFailure_ERR, FALSE);

if n == 31 then
    CheckSPAlignment();
    base = SP[];
else
    base = X[n, 64];

src = ZAvector[vec, SVL];

boolean aligned = IsAligned(base + moffs, 16);

if !aligned && AlignmentEnforced() then
    AArch64.Abort(base + moffs, AlignmentFault(accdesc));

for e = 0 to dim-1
    AArch64.MemSingle[base + moffs, 1, accdesc, aligned] = Elem[src, e,
    moffs = moffs + 1;
```

Operational information

If PSTATE.DIT is 1, the timing of this instruction is insensitive to the value of the data being loaded or stored.

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