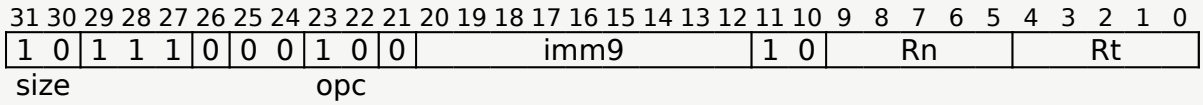
LDTRSW

Load Register Signed Word (unprivileged) loads a word from memory, sign-extends it to 64 bits, and writes the result to a register. The address that is used for the load is calculated from a base register and an immediate offset.

Memory accesses made by the instruction behave as if the instruction was executed at EL0 if the *Effective value* of PSTATE.UAO is 0 and either:

- The instruction is executed at EL1.
- The instruction is executed at EL2 when the *Effective value* of *HCR_EL2*.{E2H, TGE} is {1, 1}.

Otherwise, the memory access operates with the restrictions determined by the Exception level at which the instruction is executed. For information about memory accesses, see *Load/Store addressing modes*.




```
LDTRSW <Xt>, [<Xn|SP>{, #<simm>}]

bits(64) offset = SignExtend(imm9, 64);
```

Assembler Symbols

<Xt>	Is the 64-bit name of the general-purpose register to be transferred, encoded in the "Rt" field.
<Xn|SP>	Is the 64-bit name of the general-purpose base register or stack pointer, encoded in the "Rn" field.
<simm>	Is the optional signed immediate byte offset, in the range -256 to 255, defaulting to 0 and encoded in the "imm9" field.

Shared Decode

```
integer n = UInt(Rn);
integer t = UInt(Rt);

boolean tagchecked = n != 31;
```

Operation

```
bits(64) address;
bits(32) data;

boolean privileged = AArch64.IsUnprivAccessPriv();
```

```
AccessDescriptor accdesc = CreateAccDescGPR(MemOp_LOAD, FALSE, privileg

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

address = address + offset;

data = Mem[address, 4, accdesc];
X[t, 64] = SignExtend(data, 64);
```

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