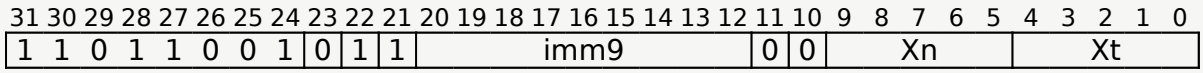
LDG

Load Allocation Tag loads an Allocation Tag from a memory address, generates a Logical Address Tag from the Allocation Tag and merges it into the destination register. The address used for the load is calculated from the base register and an immediate signed offset scaled by the Tag granule.

Integer
(FEAT_MTE)




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

```
if !IsFeatureImplemented(FEAT_MTE) then UNDEFINED;
integer t = UInt(Xt);
integer n = UInt(Xn);
bits(64) offset = LSL(SignExtend(imm9, 64), LOG2_TAG_GRANULE);
```

Assembler Symbols

<Xt>	Is the 64-bit name of the general-purpose destination register, encoded in the "Xt" field.
<Xn|SP>	Is the 64-bit name of the general-purpose base register or stack pointer, encoded in the "Xn" field.
<simm>	Is the optional signed immediate offset, a multiple of 16 in the range -4096 to 4080, defaulting to 0 and encoded in the "imm9" field.

Operation

```
bits(64) address;
bits(4) tag;

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

address = address + offset;
address = Align(address, TAG_GRANULE);

AccessDescriptor accdesc = CreateAccDescLDGSTG(MemOp_LOAD);
tag = AArch64.MemTag[address, accdesc];
X[t, 64] = AArch64.AddressWithAllocationTag(X[t, 64], tag);
```

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