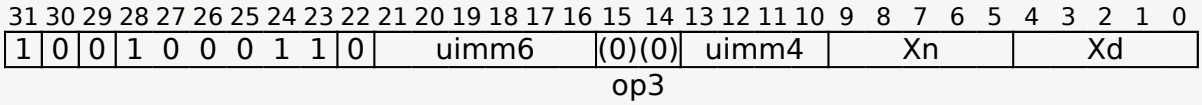
ADDG

Add with Tag adds an immediate value scaled by the Tag granule to the address in the source register, modifies the Logical Address Tag of the address using an immediate value, and writes the result to the destination register. Tags specified in GCR_EL1.Exclude are excluded from the possible outputs when modifying the Logical Address Tag.

Integer
(FEAT_MTE)



ADDG <Xd|SP>, <Xn|SP>, #<uimm6>, #<uimm4>

```
if !IsFeatureImplemented(FEAT_MTE) then UNDEFINED;
integer d = UInt(Xd);
integer n = UInt(Xn);
bits(64) offset = LSL(ZeroExtend(uimm6, 64), LOG2_TAG_GRANULE);
```

Assembler Symbols

<Xd|SP>	Is the 64-bit name of the destination general-purpose register or stack pointer, encoded in the "Xd" field.
<Xn|SP>	Is the 64-bit name of the source general-purpose register or stack pointer, encoded in the "Xn" field.
<uimm6>	Is an unsigned immediate, a multiple of 16 in the range 0 to 1008, encoded in the "uimm6" field.
<uimm4>	Is an unsigned immediate, in the range 0 to 15, encoded in the "uimm4" field.

Operation

```
bits(64) operand1 = if n == 31 then SP[] else X[n, 64];
bits(4) start_tag = AArch64.AllocationTagFromAddress(operand1);
bits(16) exclude = GCR_EL1.Exclude;
bits(64) result;
bits(4) rtag;

if AArch64.AllocationTagAccessIsEnabled(PSTATE.EL) then
    rtag = AArch64.ChooseNonExcludedTag(start_tag, uimm4, exclude);
else
    rtag = '0000';

(result, -) = AddWithCarry(operand1, offset, '0');
result = AArch64.AddressWithAllocationTag(result, rtag);
```

```
if d == 31 then
    SP[] = result;
else
    X[d, 64] = result;
```

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