IRG

Insert Random Tag inserts a random Logical Address Tag into the address in the first source register, and writes the result to the destination register. Any tags specified in the optional second source register or in GCR_EL1.Exclude are excluded from the selection of the random Logical Address Tag.

Integer
(FEAT_MTE)

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	0	0	1	1	0	1	0	1	1	0	Xm					0	0	0	1	0	0	Xn					Xd				

IRG <Xd|SP>, <Xn|SP>{, <Xm>}

```
if !IsFeatureImplemented(FEAT_MTE) then UNDEFINED;
integer d = UInt(Xd);
integer n = UInt(Xn);
integer m = UInt(Xm);
```

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 first source general-purpose register or stack pointer, encoded in the "Xn" field.

<Xm>
Is the 64-bit name of the second general-purpose source register, encoded in the "Xm" field. Defaults to XZR if absent.

Operation

```
bits(64) operand = if n == 31 then SP[] else X[n, 64];
bits(64) exclude_reg = X[m, 64];
bits(16) exclude = exclude_reg<15:0> OR GCR_EL1.Exclude;
bits(4) rtag;

if AArch64.AllocationTagAccessIsEnabled(PSTATE.EL) then
    if GCR_EL1.RRND == '1' then
        if IsOnes(exclude) then
            rtag = '0000';
        else
            rtag = ChooseRandomNonExcludedTag(exclude);
    else
        bits(4) start_tag = RGSR_EL1.TAG;
        bits(4) offset = AArch64.RandomTag();

        rtag = AArch64.ChooseNonExcludedTag(start_tag, offset, exclude);

        RGSR_EL1.TAG = rtag;
```

```
else
    rtag = '0000';

bits(64) result = AArch64.AddressWithAllocationTag(operand, rtag);

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

Base Instructions | SIMD&FP Instructions | SVE Instructions | SME Instructions | Index by Encoding | Sh Pseud

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