

DC GVA, Data Cache set Allocation Tag by VA

The DC GVA characteristics are:

Purpose

Write a value to the Allocation Tags of a naturally aligned block of N bytes, where the size of N is identified in [DCZID_EL0](#). The Allocation Tag used is determined by the input address.

Configuration

This instruction is present only when FEAT_MTE is implemented. Otherwise, direct accesses to DC GVA are undefined.

Attributes

DC GVA is a 64-bit System instruction.

Field descriptions

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| VA | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| VA | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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 |

VA, bits [63:0]

Virtual address to use. There is no alignment restriction on the address within the block of N bytes that is used.

Executing DC GVA

When this instruction is executed, it can generate memory faults or watchpoints which are prioritized in the same way as other memory-related faults or watchpoints. If a synchronous data abort fault or a watchpoint is generated, the CM bit in the ESR_ELx.ISS field is not set.

If the memory region being modified is any type of Device memory, this instruction can give an alignment fault that is prioritized in the same way as other alignment faults that are determined by the memory type.

This instruction applies to Normal memory regardless of cacheability attributes.

This instruction behaves as a set of stores to each Allocation Tag within the block being accessed, and so it:

- Generates a Permission fault if the translation system does not permit writes to the locations.
- Requires the same considerations for ordering and the management of coherency as any other store instructions.

Accesses to this instruction use the following encodings in the System instruction encoding space:

DC GVA, <Xt>

| op0 | op1 | CRn | CRm | op2 |
|------|-------|--------|--------|-------|
| 0b01 | 0b011 | 0b0111 | 0b0100 | 0b011 |

```
if PSTATE.EL == EL0 then
    if !(EL2Enabled() && HCR_EL2.<E2H,TGE> == '11')
    && SCTLR_EL1.DZE == '0' then
        if EL2Enabled() && HCR_EL2.TGE == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        else
            AArch64.SystemAccessTrap(EL1, 0x18);
        elsif EL2Enabled() && HCR_EL2.<E2H,TGE> != '11'
    && HCR_EL2.TDZ == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> != '11'
    && IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
    || SCR_EL3.FGTEn == '1') && HFGITR_EL2.DCZVA == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> == '11'
    && SCTLR_EL2.DZE == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        AArch64.MemZero(X[t, 64], CacheType_Tag);
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TDZ == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
    IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
    || SCR_EL3.FGTEn == '1') && HFGITR_EL2.DCZVA == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        AArch64.MemZero(X[t, 64], CacheType_Tag);
elseif PSTATE.EL == EL2 then
    AArch64.MemZero(X[t, 64], CacheType_Tag);
elseif PSTATE.EL == EL3 then
    AArch64.MemZero(X[t, 64], CacheType_Tag);
```

[AArch32
Registers](#)

[AArch64
Registers](#)

[AArch32
Instructions](#)

[AArch64
Instructions](#)

[Index by
Encoding](#)

[External
Registers](#)

28/03/2023 16:02; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94

Copyright © 2010-2023 Arm Limited or its affiliates. All rights reserved. This document is Non-Confidential.