TLBI VMALLE1, TLBI VMALLE1NXS, TLB Invalidate by VMID, All at stage 1, EL1

The TLBI VMALLE1, TLBI VMALLE1NXS characteristics are:

Purpose

Invalidates cached copies of translation table entries from TLBs that meet all the following requirements:

- The entry is a stage 1 translation table entry, from any level of the translation table walk.
- When EL2 is implemented and enabled in the current Security state:
  - If HCR_EL2.{E2H, TGE} is not {1, 1}, the entry would be used with the current VMID and would be required to translate the specified VA using the EL1&0 translation regime for the Security state.
  - If HCR_EL2.{E2H, TGE} is {1, 1}, the entry would be required to translate the specified VA using the EL2&0 translation regime for the Security state.
- When EL2 is not implemented or is disabled in the current Security state, the entry would be required to translate the specified VA using the EL1&0 translation regime for the Security state.

The Security state is indicated by the value of SCR_EL3.NS if FEAT_RME is not implemented, or SCR_EL3.{NSE, NS} if FEAT_RME is implemented.

The invalidation applies to the PE that executes this System instruction.

Note

For the EL1&0 translation regimes, the invalidation applies to both global entries and non-global entries with any ASID.

If FEAT_XS is implemented, the nXS variant of this System instruction is defined.

Both variants perform the same invalidation, but the TLBI System instruction without the nXS qualifier waits for all memory accesses using in-scope old translation information to complete before it is considered complete.

The TLBI System instruction with the nXS qualifier is considered complete when the subset of these memory accesses with XS attribute set to 0 are complete.

Configuration

There are no configuration notes.

Attributes

TLBI VMALLE1, TLBI VMALLE1NXS is a 64-bit System instruction.

Field descriptions

This instruction has no applicable fields.

The value in the register specified by <Xt> is ignored.

Executing TLBI VMALLE1, TLBI VMALLE1NXS

The Rt field should be set to `0b11111`. If the Rt field is not set to `0b11111`, it is constrained unpredictable whether:

- The instruction is undefined.
- The instruction behaves as if the Rt field is set to `0b11111`.

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

TLBI VMALLE1{, <Xt>}

op0	op1	CRn	CRm	op2
0b01	0b000	0b1000	0b0111	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TTLB == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
SCR_EL3.FGTEn == '1') && HFGITR_EL2.TLBIVMALLE1 ==
'1' then
```

```
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.FB == '1' then
        if IsFeatureImplemented(FEAT_XS) &&
IsFeatureImplemented(FEAT_HCX) && IsHCRXEL2Enabled()
&& HCRX_EL2.FnXS == '1' then

AArch64.TLBI_VMALL(SecurityStateAtEL(EL1),
Regime_EL10, VMID[], Shareability_ISH,
TLBI_ExcludeXS);
        else

AArch64.TLBI_VMALL(SecurityStateAtEL(EL1),
Regime_EL10, VMID[], Shareability_ISH, TLBI_AllAttr);
    else
        if IsFeatureImplemented(FEAT_XS) &&
IsFeatureImplemented(FEAT_HCX) && IsHCRXEL2Enabled()
&& HCRX_EL2.FnXS == '1' then

AArch64.TLBI_VMALL(SecurityStateAtEL(EL1),
Regime_EL10, VMID[], Shareability_NSH,
TLBI_ExcludeXS);
        else

AArch64.TLBI_VMALL(SecurityStateAtEL(EL1),
Regime_EL10, VMID[], Shareability_NSH, TLBI_AllAttr);
elsif PSTATE.EL == EL2 then
    if HCR_EL2.<E2H,TGE> == '11' then
        AArch64.TLBI_VMALL(SecurityStateAtEL(EL2),
Regime_EL20, VMID_NONE, Shareability_NSH,
TLBI_AllAttr);
    else
        AArch64.TLBI_VMALL(SecurityStateAtEL(EL1),
Regime_EL10, VMID[], Shareability_NSH, TLBI_AllAttr);
elsif PSTATE.EL == EL3 then
    if HCR_EL2.<E2H,TGE> == '11' then
        AArch64.TLBI_VMALL(SecurityStateAtEL(EL2),
Regime_EL20, VMID_NONE, Shareability_NSH,
TLBI_AllAttr);
    else
        AArch64.TLBI_VMALL(SecurityStateAtEL(EL1),
Regime_EL10, VMID[], Shareability_NSH, TLBI_AllAttr);
```

TLBI VMALLE1NXS{, <Xt>}

op0	op1	CRn	CRm	op2
0b01	0b000	0b1001	0b0111	0b000

```
if !IsFeatureImplemented(FEAT_XS) then
    UNDEFINED;
elsif PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TTLB == '1' then
```

```
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
SCR_EL3.FGTEn == '1') &&
IsFeatureImplemented(FEAT_HCX) && (!
IsHCRXEL2Enabled() || HCRX_EL2.FGTnXS == '0') &&
HFGITR_EL2.TLBIVMALLE1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.FB == '1' then
        AArch64.TLBI_VMALL(SecurityStateAtEL(EL1),
Regime_EL10, VMID[], Shareability_ISH,
TLBI_ExcludeXS);
    else
        AArch64.TLBI_VMALL(SecurityStateAtEL(EL1),
Regime_EL10, VMID[], Shareability_NSH,
TLBI_ExcludeXS);
elsif PSTATE.EL == EL2 then
    if HCR_EL2.<E2H,TGE> == '11' then
        AArch64.TLBI_VMALL(SecurityStateAtEL(EL2),
Regime_EL20, VMID_NONE, Shareability_NSH,
TLBI_ExcludeXS);
    else
        AArch64.TLBI_VMALL(SecurityStateAtEL(EL1),
Regime_EL10, VMID[], Shareability_NSH,
TLBI_ExcludeXS);
elsif PSTATE.EL == EL3 then
    if HCR_EL2.<E2H,TGE> == '11' then
        AArch64.TLBI_VMALL(SecurityStateAtEL(EL2),
Regime_EL20, VMID_NONE, Shareability_NSH,
TLBI_ExcludeXS);
    else
        AArch64.TLBI_VMALL(SecurityStateAtEL(EL1),
Regime_EL10, VMID[], Shareability_NSH,
TLBI_ExcludeXS);
```

28/03/2023 16:02; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94