

TLBI ASIDE1OS, TLBI ASIDE1OSNXS, TLB Invalidate by ASID, EL1, Outer Shareable

The TLBI ASIDE1OS, TLBI ASIDE1OSNXS 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.
- The entry would be used for the specified ASID, and either:
 - Is from a level of lookup above the final level.
 - Is a non-global entry from the final level of lookup.
- 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 an address 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 an address 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 an address 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 all PEs in the same Outer Shareable shareability domain as the PE that executes this System instruction.

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

This instruction is present only when FEAT_TLBIOS is implemented. Otherwise, direct accesses to TLBI ASIDE1OS, TLBI ASIDE1OSNXS are undefined.

Attributes

TLBI ASIDE1OS, TLBI ASIDE1OSNXS 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
ASID																RES0															
RES0																															
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

ASID, bits [63:48]

ASID value to match. Any appropriate TLB entries that match the ASID values will be affected by this System instruction.

If the implementation supports 16 bits of ASID, then the upper 8 bits of the ASID must be written to 0 by software when the context being invalidated only uses 8 bits.

Bits [47:0]

Reserved, res0.

Executing TLBI ASIDE1OS, TLBI ASIDE1OSNXS

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

TLBI ASIDE1OS{, <Xt>}

op0	op1	CRn	CRm	op2
0b01	0b000	0b1000	0b0001	0b010

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TTLB == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.TTLBOS == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
        IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
        SCR_EL3.FGTEn == '1') && HFGITR_EL2.TLBIASIDE1OS ==
        '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        if IsFeatureImplemented(FEAT_XS) &&
        IsFeatureImplemented(FEAT_HCX) && IsHCRXEL2Enabled()
        && HCRX_EL2.FnXS == '1' then

        AArch64.TLBI_ASID(SecurityStateAtEL(EL1),
        Regime_EL10, VMID[], Shareability_OSH,
        TLBI_ExcludeXS, X[t, 64]);
        else

        AArch64.TLBI_ASID(SecurityStateAtEL(EL1),
        Regime_EL10, VMID[], Shareability_OSH, TLBI_AllAttr,
        X[t, 64]);
    elsif PSTATE.EL == EL2 then
        if HCR_EL2.<E2H,TGE> == '11' then
            AArch64.TLBI_ASID(SecurityStateAtEL(EL2),
            Regime_EL20, VMID_NONE, Shareability_OSH,
            TLBI_AllAttr, X[t, 64]);
        else
            AArch64.TLBI_ASID(SecurityStateAtEL(EL1),
            Regime_EL10, VMID[], Shareability_OSH, TLBI_AllAttr,
            X[t, 64]);
    elsif PSTATE.EL == EL3 then
        if HCR_EL2.<E2H,TGE> == '11' then
            AArch64.TLBI_ASID(SecurityStateAtEL(EL2),
            Regime_EL20, VMID_NONE, Shareability_OSH,
            TLBI_AllAttr, X[t, 64]);
        else
            AArch64.TLBI_ASID(SecurityStateAtEL(EL1),
            Regime_EL10, VMID[], Shareability_OSH, TLBI_AllAttr,
            X[t, 64]);

```

TLBI ASIDE1OSNXS{, <Xt>}

op0	op1	CRn	CRm	op2
0b01	0b000	0b1001	0b0001	0b010

```

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() && HCR_EL2.TTLBOS == '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.TLBIASIDE1OS == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        else
            AArch64.TLBI_ASID(SecurityStateAtEL(EL1),
            Regime_EL10, VMID[], Shareability_OSH,
            TLBI_ExcludeXS, X[t, 64]);
        elsif PSTATE.EL == EL2 then
            if HCR_EL2.<E2H,TGE> == '11' then
                AArch64.TLBI_ASID(SecurityStateAtEL(EL2),
                Regime_EL20, VMID_NONE, Shareability_OSH,
                TLBI_ExcludeXS, X[t, 64]);
            else
                AArch64.TLBI_ASID(SecurityStateAtEL(EL1),
                Regime_EL10, VMID[], Shareability_OSH,
                TLBI_ExcludeXS, X[t, 64]);
            elsif PSTATE.EL == EL3 then
                if HCR_EL2.<E2H,TGE> == '11' then
                    AArch64.TLBI_ASID(SecurityStateAtEL(EL2),
                    Regime_EL20, VMID_NONE, Shareability_OSH,
                    TLBI_ExcludeXS, X[t, 64]);
                else
                    AArch64.TLBI_ASID(SecurityStateAtEL(EL1),
                    Regime_EL10, VMID[], Shareability_OSH,
                    TLBI_ExcludeXS, X[t, 64]);
            end
        end
    end
end

```

[AArch32
Registers](#)

[AArch64
Registers](#)

[AArch32
Instructions](#)

[AArch64
Instructions](#)

[Index by
Encoding](#)

[External
Registers](#)

28/03/2023 16:02; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94

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