TTBR0_EL1, Translation Table Base Register 0 (EL1)

The TTBR0_EL1 characteristics are:

Purpose

Holds the base address of the translation table for the initial lookup for stage 1 of the translation of an address from the lower VA range in the EL1&0 translation regime, and other information for this translation regime.

Configuration

AArch64 System register TTBR0_EL1 bits [63:0] are architecturally mapped to AArch32 System register TTBR0[63:0].

TTBR0_EL1 is a 128-bit register that can also be accessed as a 64-bit value. If it is accessed as a 64-bit register, accesses read and write bits [63:0] and do not modify bits [127:64].

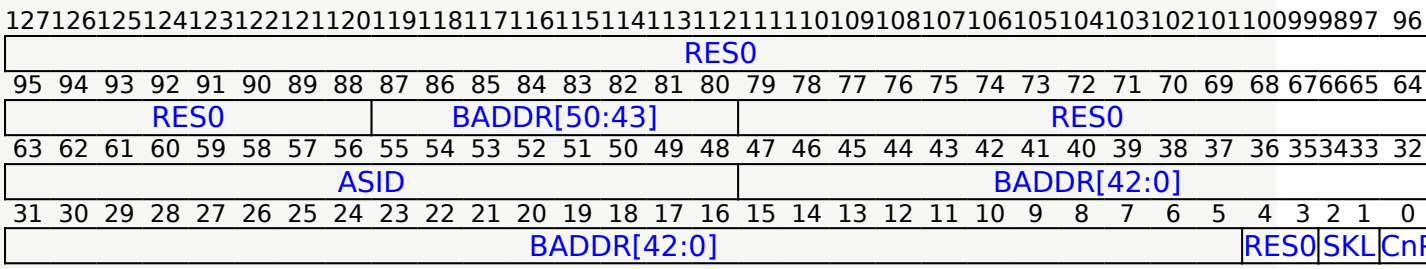
Attributes

TTBR0_EL1 is a:

- 128-bit register when FEAT_D128 is implemented and TCR2_EL1.D128 == 1
- 64-bit register when FEAT_D128 is not implemented or TCR2_EL1.D128 == 0

Field descriptions

When FEAT_D128 is implemented and TCR2_EL1.D128 == 1:



Bits [127:88]

Reserved, res0.

BADDR, bits [87:80, 47:5]

Translation table base address:

- Bits A[55:x] of the stage 1 translation table base address bits are in register bits[87:80, 47:x].
- Bits A[(x-1):0] of the stage 1 translation table base address are zero.

Address bit x is the minimum address bit required to align the translation table to the size of the table. x is calculated based on LOG2(StartTableSize), as described in VMSAv9-128. The smallest permitted value of x is 5.

The BADDR field is split as follows:

- BADDR[50:43] is TTBR0_EL1[87:80].
- BADDR[42:0] is TTBR0_EL1[47:5].

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

Bits [79:64]

Reserved, res0.

ASID, bits [63:48]

An ASID for the translation table base address. The TCR_EL1.A1 field selects either TTBR0_EL1.ASID or TTBR1_EL1.ASID.

If the implementation has only 8 bits of ASID, then the upper 8 bits of this field are res0.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

Bits [4:3]

Reserved, res0.

SKL, bits [2:1]

Skip Level associated with translation table walks using TTBR0_EL1.

This determines the number of levels to be skipped from the regular start level of the Stage 1 EL1&0 translation table walks using TTBR0_EL1.

SKL	Meaning
0b00	Skip 0 level from the regular start level.
0b01	Skip 1 level from the regular start level.
0b10	Skip 2 levels from the regular start level.
0b11	Skip 3 levels from the regular start level.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

CnP, bit [0]
When FEAT_TTCNP is implemented:

Common not Private. This bit indicates whether each entry that is pointed to by TTBR0_EL1 is a member of a common set that can be used by every PE in the Inner Shareable domain for which the value of TTBR0_EL1.CnP is 1.

CnP	Meaning
0b0	The translation table entries pointed to by TTBR0_EL1, for the current translation regime and ASID, are permitted to differ from corresponding entries for TTBR0_EL1 for other PEs in the Inner Shareable domain. This is not affected by: • The value of TTBR0_EL1.CnP on those other PEs. • The value of the current ASID. • If EL2 is implemented and enabled in the current Security state, the value of the current VMID.

0b1	The translation table entries pointed to by TTBR0_EL1 are the same as the translation table entries for every other PE in the Inner Shareable domain for which the value of TTBR0_EL1.CnP is 1 and all of the following apply: • The translation table entries are pointed to by TTBR0_EL1. • The translation tables relate to the same translation regime. • The ASID is the same as the current ASID. • If EL2 is implemented and enabled in the current Security state, the value of the current VMID.

This bit is permitted to be cached in a TLB.

When a TLB combines entries from stage 1 translation and stage 2 translation into a single entry, that entry can only be shared between different PEs if the value of the CnP bit is 1 for both stage 1 and stage 2.

Note

If the value of the TTBR0_EL1.CnP bit is 1 on multiple PEs in the same Inner Shareable domain and those TTBR0_EL1s do not point to the same translation table entries when the other conditions specified for the case when the value of CnP is 1 apply, then the results of translations are constrained unpredictable, see 'CONSTRAINED UNPREDICTABLE behaviors due to caching of control or data values'.

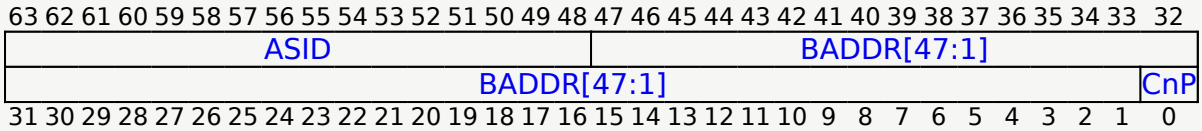
The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

Otherwise:

Reserved, res0.

When FEAT_D128 is not implemented or TCR2_EL1.D128 == 0:



ASID, bits [63:48]

An ASID for the translation table base address. The TCR_EL1.A1 field selects either TTBR0_EL1.ASID or TTBR1_EL1.ASID.

If the implementation has only 8 bits of ASID, then the upper 8 bits of this field are res0.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

BADDR[47:1], bits [47:1]

Translation table base address:

- Bits A[47:x] of the stage 1 translation table base address bits are in register bits[47:x].
- Bits A[(x-1):0] of the stage 1 translation table base address are zero.

Address bit x is the minimum address bit required to align the translation table to the size of the table. The AArch64 Virtual Memory System Architecture chapter describes how x is calculated based on the value of TCR_EL1.T0SZ, the translation stage, and the translation granule size.

Note

If an OA size of more than 48 bits is in use, and the translation table has fewer than eight entries, the table must be aligned to 64 bytes. Otherwise the translation table must be aligned to the size of the table.

If the value of TCR_EL1.IPS is not `0b110`, then:

- Register bits[(x-1):1] are res0.

- If the implementation supports 52-bit PAs and IPAs, then bits A[51:48] of the stage 1 translation table base address are `0b0000`.

If FEAT_LPA is implemented and the value of TCR_EL1.IPS is `0b110`, then:

- Bits A[51:48] of the stage 1 translation table base address bits are in register bits[5:2].
- Register bit[1] is res0.
- The smallest permitted value of x is 6.
- When x>6, register bits[(x-1):6] are res0.

Note

TCR_EL1.IPS==`0b110` is permitted when:

- FEAT_LPA is implemented and the 64KB translation granule is used.
- FEAT_LPA2 is implemented and the 4KB or 16KB translation granule is used.

When the value of ID_AA64MMFR0_EL1.PARange indicates that the implementation does not support a 52 bit PA size, if a translation table lookup uses this register when the Effective value of TCR_EL1.IPS is `0b110` and the value of register bits[5:2] is nonzero, an Address size fault is generated.

When the value of ID_AA64MMFR0_EL1.PARange indicates that the implementation supports a 56 bit PA size, bits A[55:52] of the stage 1 translation table base address are zero.

If any register bit[47:1] that is defined as res0 has the value 1 when a translation table walk is done using TTBR0_EL1, then the translation table base address might be misaligned, with effects that are constrained unpredictable, and must be one of the following:

- Bits A[(x-1):0] of the stage 1 translation table base address are treated as if all the bits are zero. The value read back from the corresponding register bits is either the value written to the register or zero.
- The result of the calculation of an address for a translation table walk using this register can be corrupted in those bits that are nonzero.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

CnP, bit [0]
When FEAT_TTCNP is implemented:

Common not Private. This bit indicates whether each entry that is pointed to by TTBR0_EL1 is a member of a common set that can be used by every PE in the Inner Shareable domain for which the value of TTBR0_EL1.CnP is 1.

CnP	Meaning
0b0	The translation table entries pointed to by TTBR0_EL1, for the current translation regime and ASID, are permitted to differ from corresponding entries for TTBR0_EL1 for other PEs in the Inner Shareable domain. This is not affected by: • The value of TTBR0_EL1.CnP on those other PEs. • The value of the current ASID. • If EL2 is implemented and enabled in the current Security state, the value of the current VMID.

0b1	The translation table entries pointed to by TTBR0_EL1 are the same as the translation table entries for every other PE in the Inner Shareable domain for which the value of TTBR0_EL1.CnP is 1 and all of the following apply: • The translation table entries are pointed to by TTBR0_EL1. • The translation tables relate to the same translation regime. • The ASID is the same as the current ASID. • If EL2 is implemented and enabled in the current Security state, the value of the current VMID.

This bit is permitted to be cached in a TLB.

When a TLB combines entries from stage 1 translation and stage 2 translation into a single entry, that entry can only be shared between different PEs if the value of the CnP bit is 1 for both stage 1 and stage 2.

Note

If the value of the TTBR0_EL1.CnP bit is 1 on multiple PEs in the same Inner Shareable domain and those TTBR0_EL1s do not point to the same translation table entries when the other conditions specified for the case when the value of CnP is 1 apply, then the results of translations are constrained unpredictable, see 'CONSTRAINED UNPREDICTABLE behaviors due to caching of control or data values'.

The reset behavior of this field is:

- On a Warm reset, this field resets to an architecturally unknown value.

Otherwise:

Reserved, res0.

Accessing TTBR0_EL1

When HCR_EL2.E2H is 1, without explicit synchronization, access from EL3 using the mnemonic TTBR0_EL1 or TTBR0_EL12 are not guaranteed to be ordered with respect to accesses using the other mnemonic.

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

MRS <Xt>, TTBR0_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0010	0b0000	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TRVM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
SCR_EL3.FGTEn == '1') && HFGRTR_EL2.TTBR0_EL1 == '1'
then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<NV2,NV1,NV> ==
'111' then
        X[t, 64] = NVMem[0x200];
    else
        X[t, 64] = TTBR0_EL1<63:0>;
elsif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' then
        X[t, 64] = TTBR0_EL2<63:0>;
    else
        X[t, 64] = TTBR0_EL1<63:0>;
elsif PSTATE.EL == EL3 then
    X[t, 64] = TTBR0_EL1<63:0>;
```

MSR TTBR0_EL1, <Xt>

op0	op1	CRn	CRm	op2

0b11	0b000	0b0010	0b0000	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TVM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
SCR_EL3.FGTEn == '1') && HFGWTR_EL2.TTBR0_EL1 == '1'
then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<NV2,NV1,NV> ==
'111' then
        NVMem[0x200] = X[t, 64];
    else
        TTBR0_EL1<63:0> = X[t, 64];
elsif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' then
        TTBR0_EL2<63:0> = X[t, 64];
    else
        TTBR0_EL1<63:0> = X[t, 64];
elsif PSTATE.EL == EL3 then
    TTBR0_EL1<63:0> = X[t, 64];
```

MRS <Xt>, TTBR0_EL12

op0	op1	CRn	CRm	op2
0b11	0b101	0b0010	0b0000	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV1,NV> == '101'
then
        X[t, 64] = NVMem[0x200];
    elsif EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' then
        X[t, 64] = TTBR0_EL1<63:0>;
    else
        UNDEFINED;
elsif PSTATE.EL == EL3 then
    if EL2Enabled() && !ELUsingAArch32(EL2) &&
HCR_EL2.E2H == '1' then
        X[t, 64] = TTBR0_EL1<63:0>;
    else
        UNDEFINED;
```

MSR TTBR0_EL12, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b101	0b0010	0b0000	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV1,NV> == '101'
then
        NVMem[0x200] = X[t, 64];
    elsif EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' then
        TTBR0_EL1<63:0> = X[t, 64];
    else
        UNDEFINED;
elsif PSTATE.EL == EL3 then
    if EL2Enabled() && !ELUsingAArch32(EL2) &&
HCR_EL2.E2H == '1' then
        TTBR0_EL1<63:0> = X[t, 64];
    else
        UNDEFINED;
```

When FEAT_D128 is implemented

MRRS <Xt+1>, <Xt>, TTBR0_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0010	0b0000	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && SCR_EL3.D128En == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TRVM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x14);
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
SCR_EL3.FGTEn == '1') && HFGRTR_EL2.TTBR0_EL1 == '1'
then
        AArch64.SystemAccessTrap(EL2, 0x14);
    elsif EL2Enabled() && (!IsHCRXEL2Enabled() ||
```

```
HCRX_EL2.D128En == '0') then
        AArch64.SystemAccessTrap(EL2, 0x14);
    elsif HaveEL(EL3) && SCR_EL3.D128En == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x14);
    elsif EL2Enabled() && HCR_EL2.<NV2,NV1,NV> ==
'111' then
        (X[t + 1, 64], X[t, 64]) = (NVMem[0x208],
NVMem[0x200]);
    else
        (X[t + 1, 64], X[t, 64]) =
(TTBR0_EL1<127:64>, TTBR0_EL1<63:0>);
elsif PSTATE.EL == EL2 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && SCR_EL3.D128En == '0' then
        UNDEFINED;
    elsif HaveEL(EL3) && SCR_EL3.D128En == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x14);
    elsif HCR_EL2.E2H == '1' then
        (X[t + 1, 64], X[t, 64]) =
(TTBR0_EL2<127:64>, TTBR0_EL2<63:0>);
    else
        (X[t + 1, 64], X[t, 64]) =
(TTBR0_EL1<127:64>, TTBR0_EL1<63:0>);
elsif PSTATE.EL == EL3 then
    (X[t + 1, 64], X[t, 64]) = (TTBR0_EL1<127:64>,
TTBR0_EL1<63:0>);
```

When FEAT_D128 is implemented

MSRR TTBR0_EL1, <Xt+1>, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0010	0b0000	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && SCR_EL3.D128En == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TVM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x14);
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
SCR_EL3.FGTEn == '1') && HFGWTR_EL2.TTBR0_EL1 == '1'
```

```
then
        AArch64.SystemAccessTrap(EL2, 0x14);
    elsif EL2Enabled() && (!IsHCRXEL2Enabled() ||
HCRX_EL2.D128En == '0') then
        AArch64.SystemAccessTrap(EL2, 0x14);
    elsif HaveEL(EL3) && SCR_EL3.D128En == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x14);
    elsif EL2Enabled() && HCR_EL2.<NV2,NV1,NV> ==
'111' then
        (NVMem[0x208], NVMem[0x200]) = (X[t + 1,
64], X[t, 64]);
    else
        (TTBR0_EL1<127:64>, TTBR0_EL1<63:0>) = (X[t
+ 1, 64], X[t, 64]);
elsif PSTATE.EL == EL2 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && SCR_EL3.D128En == '0' then
        UNDEFINED;
    elsif HaveEL(EL3) && SCR_EL3.D128En == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x14);
    elsif HCR_EL2.E2H == '1' then
        (TTBR0_EL2<127:64>, TTBR0_EL2<63:0>) = (X[t
+ 1, 64], X[t, 64]);
    else
        (TTBR0_EL1<127:64>, TTBR0_EL1<63:0>) = (X[t
+ 1, 64], X[t, 64]);
elsif PSTATE.EL == EL3 then
    (TTBR0_EL1<127:64>, TTBR0_EL1<63:0>) = (X[t + 1,
64], X[t, 64]);
```

When FEAT_D128 is implemented

MRRS <Xt+1>, <Xt>, TTBR0_EL12

op0	op1	CRn	CRm	op2
0b11	0b101	0b0010	0b0000	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV1,NV> == '101'
then
        (X[t + 1, 64], X[t, 64]) = (NVMem[0x208],
NVMem[0x200]);
    elsif EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
```

```
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' then
        if Halted() && HaveEL(EL3) && EDSCR.SDD ==
'1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && SCR_EL3.D128En == '0'
then
            UNDEFINED;
        elsif HaveEL(EL3) && SCR_EL3.D128En == '0'
then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x14);
        else
            (X[t + 1, 64], X[t, 64]) =
(TTBR0_EL1<127:64>, TTBR0_EL1<63:0>);
    else
        UNDEFINED;
elsif PSTATE.EL == EL3 then
    if EL2Enabled() && !ELUsingAArch32(EL2) &&
HCR_EL2.E2H == '1' then
        (X[t + 1, 64], X[t, 64]) =
(TTBR0_EL1<127:64>, TTBR0_EL1<63:0>);
    else
        UNDEFINED;
```

When FEAT_D128 is implemented

MSRR TTBR0_EL12, <Xt+1>, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b101	0b0010	0b0000	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV1,NV> == '101'
then
        (NVMem[0x208], NVMem[0x200]) = (X[t + 1,
64], X[t, 64]);
    elsif EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' then
        if Halted() && HaveEL(EL3) && EDSCR.SDD ==
'1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && SCR_EL3.D128En == '0'
then
            UNDEFINED;
```

```
        elsif HaveEL(EL3) && SCR_EL3.D128En == '0'
then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
        else
            (TTBR0_EL1<127:64>, TTBR0_EL1<63:0>) =
(X[t + 1, 64], X[t, 64]);
    else
        UNDEFINED;
elsif PSTATE.EL == EL3 then
    if EL2Enabled() && !ELUsingAArch32(EL2) &&
HCR_EL2.E2H == '1' then
        (TTBR0_EL1<127:64>, TTBR0_EL1<63:0>) = (X[t
+ 1, 64], X[t, 64]);
    else
        UNDEFINED;
```

28/03/2023 16:02; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94