

# TRCCCCTLR, Cycle Count Control Register

The TRCCCCTLR characteristics are:

## Purpose

Set the threshold value for cycle counting.

## Configuration

AArch64 System register TRCCCCTLR bits [31:0] are architecturally mapped to External register [TRCCCCTLR\[31:0\]](#).

This register is present only when FEAT\_ETE is implemented, FEAT\_TRC\_SR is implemented and TRCIDR0.TRCCCI == 1. Otherwise, direct accesses to TRCCCCTLR are undefined.

## Attributes

TRCCCCTLR is a 64-bit register.

## 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 |
| RES0 |    |    |    |    |    |    |    |    |    |    |    |           |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| RES0 |    |    |    |    |    |    |    |    |    |    |    | THRESHOLD |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 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  |

### Bits [63:12]

Reserved, res0.

### THRESHOLD, bits [11:0]

Sets the threshold value for instruction trace cycle counting.

The minimum threshold value that can be programmed into THRESHOLD is given in [TRCIDR3.CCITMIN](#). If the THRESHOLD value is smaller than the value in [TRCIDR3.CCITMIN](#) then the behavior is constrained unpredictable. That is, cycle counts might or might not be included in the trace and the cycle count threshold is not known.

Writing a value of zero when [TRCCONFIGR](#).CCI enables instruction trace cycle counting results in constrained unpredictable behavior. That is, cycle counts might or might not be included in the trace and the cycle count threshold is not known.

The reset behavior of this field is:

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

## Accessing TRCCCCTLR

Must be programmed if [TRCCONFIGR](#).CCI == 1.

Writes are constrained unpredictable if the trace unit is not in the Idle state.

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

### MRS <Xt>, TRCCCCTLR

| op0  | op1   | CRn    | CRm    | op2   |
|------|-------|--------|--------|-------|
| 0b10 | 0b001 | 0b0000 | 0b1110 | 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'" && CPTR_EL3.TTA == '1' then
        UNDEFINED;
    elsif CPACR_EL1.TTA == '1' then
        AArch64.SystemAccessTrap(EL1, 0x18);
    elsif EL2Enabled() && CPTR_EL2.TTA == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
    IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
    SCR_EL3.FGTEn == '1') && HDFGRTR_EL2.TRC == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif HaveEL(EL3) && CPTR_EL3.TTA == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            X[t, 64] = TRCCCCTLR;
    elsif PSTATE.EL == EL2 then
        if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
        && boolean IMPLEMENTATION_DEFINED "EL3 trap priority
        when SDD == '1'" && CPTR_EL3.TTA == '1' then
            UNDEFINED;
```

```

    elsif CPTR_EL2.TTA == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif HaveEL(EL3) && CPTR_EL3.TTA == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            X[t, 64] = TRCCCCTLR;
    elsif PSTATE.EL == EL3 then
        if CPTR_EL3.TTA == '1' then
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            X[t, 64] = TRCCCCTLR;

```

## MSR TRCCCCTLR, <Xt>

| op0  | op1   | CRn    | CRm    | op2   |
|------|-------|--------|--------|-------|
| 0b10 | 0b001 | 0b0000 | 0b1110 | 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'" && CPTR_EL3.TTA == '1' then
        UNDEFINED;
    elsif CPACR_EL1.TTA == '1' then
        AArch64.SystemAccessTrap(EL1, 0x18);
    elsif EL2Enabled() && CPTR_EL2.TTA == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
    IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
    SCR_EL3.FGTEn == '1') && HDFGWTR_EL2.TRC == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif HaveEL(EL3) && CPTR_EL3.TTA == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            TRCCCCTLR = X[t, 64];
    elsif PSTATE.EL == EL2 then
        if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
        && boolean IMPLEMENTATION_DEFINED "EL3 trap priority
        when SDD == '1'" && CPTR_EL3.TTA == '1' then
            UNDEFINED;
        elsif CPTR_EL2.TTA == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        elsif HaveEL(EL3) && CPTR_EL3.TTA == '1' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else

```

```
        AArch64.SystemAccessTrap(EL3, 0x18);
    else
        TRCCCCTLR = X[t, 64];
elseif PSTATE.EL == EL3 then
    if CPTR_EL3.TTA == '1' then
        AArch64.SystemAccessTrap(EL3, 0x18);
    else
        TRCCCCTLR = X[t, 64];
```

---

[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.