

TRCCLAIMSET, Claim Tag Set Register

The TRCCLAIMSET characteristics are:

Purpose

In conjunction with [TRCCLAIMCLR](#), provides Claim Tag bits that can be separately set and cleared to indicate whether functionality is in use by a debug agent.

For additional information, see the CoreSight Architecture Specification.

Configuration

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

This register is present only when FEAT_ETE is implemented and FEAT_TRC_SR is implemented. Otherwise, direct accesses to TRCCLAIMSET are undefined.

The number of claim tag bits implemented is implementation defined. Arm recommends that implementations support a minimum of four claim tag bits, that is, SET[3:0] reads as 0b1111.

Attributes

TRCCLAIMSET 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	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
SET[31]	SET[30]	SET[29]	SET[28]	SET[27]	SET[26]	SET[25]	SET[24]	SET[23]	SET[22]	SET[21]	SET[20]	SET[19]	SET[18]	SET[17]	SET[16]	SET[15]	SET[14]	SET[13]	SET[12]	SET[11]	SET[10]	SET[9]	SET[8]	SET[7]	SET[6]	SET[5]	SET[4]	SET[3]	SET[2]	SET[1]	SET[0]	SET[31]	SET[30]	SET[29]	SET[28]	SET[27]	SET[26]	SET[25]	SET[24]	SET[23]	SET[22]	SET[21]	SET[20]	SET[19]	SET[18]	SET[17]	SET[16]	SET[15]	SET[14]	SET[13]	SET[12]	SET[11]	SET[10]	SET[9]	SET[8]	SET[7]	SET[6]	SET[5]	SET[4]	SET[3]	SET[2]	SET[1]	SET[0]

Bits [63:32]

Reserved, res0.

SET[<m>], bit [m], for m = 31 to 0

Claim Tag Set. Indicates whether Claim Tag bit <m> is implemented, and is used to set Claim Tag bit <m> to 1.

SET[<m>]	Meaning
----------	---------

0b0	On a read: Claim Tag bit <m> is not implemented. On a write: Ignored.
0b1	On a read: Claim Tag bit <m> is implemented. On a write: Set Claim Tag bit <m> to 1.

This bit reads-as-zero and ignores writes if m > the number of Claim Tag bits.

Access to this field is **RAO/W1S**.

Accessing TRCCLAIMSET

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

MRS <Xt>, TRCCLAIMSET

op0	op1	CRn	CRm	op2
0b10	0b001	0b0111	0b1000	0b110

```

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.TRCCLAIM == '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] = TRCCLAIMSET;
    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;

```

```

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

```

MSR TRCCLAIMSET, <Xt>

op0	op1	CRn	CRm	op2
0b10	0b001	0b0111	0b1000	0b110

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif 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;
    elseif CPACR_EL1.TTA == '1' then
        AArch64.SystemAccessTrap(EL1, 0x18);
    elseif EL2Enabled() && CPTR_EL2.TTA == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() &&
    IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
    SCR_EL3.FGTEn == '1') && HDFGWTR_EL2.TRCLAIM == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif HaveEL(EL3) && CPTR_EL3.TTA == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            TRCCLAIMSET = X[t, 64];
elseif 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;
    elseif CPTR_EL2.TTA == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif HaveEL(EL3) && CPTR_EL3.TTA == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;

```

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

[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.