

CNTP_CVAL_EL0, Counter-timer Physical Timer CompareValue Register

The CNTP_CVAL_EL0 characteristics are:

Purpose

Holds the compare value for the EL1 physical timer.

Configuration

AArch64 System register CNTP_CVAL_EL0 bits [63:0] are architecturally mapped to AArch32 System register [CNTP_CVAL\[63:0\]](#).

Attributes

CNTP_CVAL_EL0 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
CompareValue																															
CompareValue																															
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

CompareValue, bits [63:0]

Holds the EL1 physical timer CompareValue.

When [CNTP_CTL_EL0](#).ENABLE is 1, the timer condition is met when ([CNTPCT_EL0](#) - CompareValue) is greater than or equal to zero. This means that CompareValue acts like a 64-bit upcounter timer. When the timer condition is met:

- [CNTP_CTL_EL0](#).ISTATUS is set to 1.
- If [CNTP_CTL_EL0](#).IMASK is 0, an interrupt is generated.

When [CNTP_CTL_EL0](#).ENABLE is 0, the timer condition is not met, but [CNTPCT_EL0](#) continues to count.

If the Generic counter is implemented at a size less than 64 bits, then this field is permitted to be implemented at the same width as the counter, and the upper bits are res0.

The value of this field is treated as zero-extended in all counter calculations.

The reset behavior of this field is:

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

Accessing CNTP_CVAL_EL0

When [HCR_EL2.E2H](#) is 1, without explicit synchronization, access from EL3 using the mnemonic CNTP_CVAL_EL0 or CNTP_CVAL_EL02 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>, CNTP_CVAL_EL0

op0	op1	CRn	CRm	op2
0b11	0b011	0b1110	0b0010	0b010

```
if PSTATE.EL == EL0 then
    if !(EL2Enabled() && HCR_EL2.<E2H,TGE> == '11')
    && CNTKCTL_EL1.EL0PTEN == '0' then
        if EL2Enabled() && HCR_EL2.TGE == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        else
            AArch64.SystemAccessTrap(EL1, 0x18);
        elsif EL2Enabled() && HCR_EL2.E2H == '0' &&
        CNTHCTL_EL2.EL1PCEN == '0' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        elsif EL2Enabled() && HCR_EL2.<E2H,TGE> == '10'
        && CNTHCTL_EL2.EL1PTEN == '0' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        elsif EL2Enabled() && HCR_EL2.<E2H,TGE> == '11'
        && CNTHCTL_EL2.EL0PTEN == '0' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        elsif EL2Enabled() && HCR_EL2.<E2H,TGE> == '11'
        && SCR_EL3.NS == '0' &&
        IsFeatureImplemented(FEAT_SEL2) then
            X[t, 64] = CNTHPS_CVAL_EL2;
        elsif EL2Enabled() && HCR_EL2.<E2H,TGE> == '11'
        && SCR_EL3.NS == '1' then
            X[t, 64] = CNTHP_CVAL_EL2;
        else
            X[t, 64] = CNTP_CVAL_EL0;
    elsif PSTATE.EL == EL1 then
        if EL2Enabled() && HCR_EL2.E2H == '0' &&
        CNTHCTL_EL2.EL1PCEN == '0' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        elsif EL2Enabled() && HCR_EL2.E2H == '1' &&
        CNTHCTL_EL2.EL1PTEN == '0' then
            AArch64.SystemAccessTrap(EL2, 0x18);
```

```

        elsif EL2Enabled() && HCR_EL2.<NV2,NV1,NV> ==
'111' then
            X[t, 64] = NVMem[0x178];
        else
            X[t, 64] = CNTP_CVAL_EL0;
    elsif PSTATE.EL == EL2 then
        if HCR_EL2.E2H == '1' && SCR_EL3.NS == '0' &&
IsFeatureImplemented(FEAT_SEL2) then
            X[t, 64] = CNTHPS_CVAL_EL2;
        elsif HCR_EL2.E2H == '1' && SCR_EL3.NS == '1'
then
            X[t, 64] = CNTHP_CVAL_EL2;
        else
            X[t, 64] = CNTP_CVAL_EL0;
    elsif PSTATE.EL == EL3 then
        X[t, 64] = CNTP_CVAL_EL0;

```

MSR CNTP_CVAL_EL0, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b011	0b1110	0b0010	0b010

```

if PSTATE.EL == EL0 then
    if !(EL2Enabled() && HCR_EL2.<E2H,TGE> == '11')
&& CNTKCTL_EL1.EL0PTEN == '0' then
        if EL2Enabled() && HCR_EL2.TGE == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        else
            AArch64.SystemAccessTrap(EL1, 0x18);
    elsif EL2Enabled() && HCR_EL2.E2H == '0' &&
CNTHCTL_EL2.EL1PCEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> == '10'
&& CNTHCTL_EL2.EL1PTEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> == '11'
&& CNTHCTL_EL2.EL0PTEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> == '11'
&& SCR_EL3.NS == '0' &&
IsFeatureImplemented(FEAT_SEL2) then
        CNTHPS_CVAL_EL2 = X[t, 64];
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> == '11'
&& SCR_EL3.NS == '1' then
        CNTHP_CVAL_EL2 = X[t, 64];
    else
        CNTP_CVAL_EL0 = X[t, 64];
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.E2H == '0' &&
CNTHCTL_EL2.EL1PCEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.E2H == '1' &&
CNTHCTL_EL2.EL1PTEN == '0' then

```

```

        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && HCR_EL2.<NV2,NV1,NV> ==
'111' then
        NVMem[0x178] = X[t, 64];
    else
        CNTP_CVAL_EL0 = X[t, 64];
elseif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' && SCR_EL3.NS == '0' &&
IsFeatureImplemented(FEAT_SEL2) then
        CNTHPS_CVAL_EL2 = X[t, 64];
    elseif HCR_EL2.E2H == '1' && SCR_EL3.NS == '1'
then
        CNTHP_CVAL_EL2 = X[t, 64];
    else
        CNTP_CVAL_EL0 = X[t, 64];
elseif PSTATE.EL == EL3 then
    CNTP_CVAL_EL0 = X[t, 64];

```

MRS <Xt>, CNTP_CVAL_EL02

op0	op1	CRn	CRm	op2
0b11	0b101	0b1110	0b0010	0b010

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV1,NV> == '101'
then
        if EL2Enabled() && HCR_EL2.<E2H,TGE> != '11'
&& CNTHCTL_EL2.EL1NVPCT == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        else
            X[t, 64] = NVMem[0x178];
    elseif EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elseif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' then
        X[t, 64] = CNTP_CVAL_EL0;
    else
        UNDEFINED;
elseif PSTATE.EL == EL3 then
    if EL2Enabled() && !ELUsingAArch32(EL2) &&
HCR_EL2.E2H == '1' then
        X[t, 64] = CNTP_CVAL_EL0;
    else
        UNDEFINED;

```

MSR CNTP_CVAL_EL02, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b101	0b1110	0b0010	0b010

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV1,NV> == '101'
    then
        if EL2Enabled() && HCR_EL2.<E2H,TGE> != '11'
        && CNTHCTL_EL2.EL1NVPCT == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        else
            NVMem[0x178] = 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
            CNTP_CVAL_EL0 = X[t, 64];
        else
            UNDEFINED;
    elsif PSTATE.EL == EL3 then
        if EL2Enabled() && !ELUsingAArch32(EL2) &&
        HCR_EL2.E2H == '1' then
            CNTP_CVAL_EL0 = X[t, 64];
        else
            UNDEFINED;
```

[AArch32
Registers](#)[AArch64
Registers](#)[AArch32
Instructions](#)[AArch64
Instructions](#)[Index by
Encoding](#)[External
Registers](#)

28/03/2023 16:01; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94

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