CNTPS_TVAL_EL1, Counter-timer Physical Secure Timer TimerValue register

The CNTPS_TVAL_EL1 characteristics are:

Purpose

Holds the timer value for the secure physical timer, usually accessible at EL3 but configurably accessible at EL1 in Secure state.

Configuration

This register is present only when EL3 is implemented. Otherwise, direct accesses to CNTPS_TVAL_EL1 are undefined.

Attributes

CNTPS_TVAL_EL1 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																															
TimerValue																															
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 [63:32]

Reserved, res0.

TimerValue, bits [31:0]

The TimerValue view of the secure physical timer.

On a read of this register:

- If CNTPS_CTL_EL1.ENABLE is 0, the value returned is unknown.
- If CNTPS_CTL_EL1.ENABLE is 1, the value returned is (CNTPS_CVAL_EL1 - CNTPCT_EL0).

On a write of this register, CNTPS_CVAL_EL1 is set to (CNTPCT_EL0 + TimerValue), where TimerValue is treated as a signed 32-bit integer.

When CNTPS_CTL_EL1.ENABLE is 1, the timer condition is met when (CNTPCT_EL0 - CNTPS_CVAL_EL1) is greater than or equal to

zero. This means that TimerValue acts like a 32-bit downcounter timer. When the timer condition is met:

- CNTPS_CTL_EL1.ISTATUS is set to 1.
- If CNTPS_CTL_EL1.IMASK is 0, an interrupt is generated.

When CNTPS_CTL_EL1.ENABLE is 0, the timer condition is not met, but CNTPCT_EL0 continues to count, so the TimerValue view appears to continue to count down.

The reset behavior of this field is:

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

Accessing CNTPS_TVAL_EL1

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

MRS <Xt>, CNTPS_TVAL_EL1

op0	op1	CRn	CRm	op2
0b11	0b111	0b1110	0b0010	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if HaveEL(EL3) && SCR_EL3.NS == '0' then
        if SCR_EL3.EEL2 == '1' then
            UNDEFINED;
        elsif SCR_EL3.ST == '0' then
            AArch64.SystemAccessTrap(EL3, 0x18);
        elsif IsFeatureImplemented(FEAT_ECV) &&
EL2Enabled() && SCR_EL3.ECVEn == '1' &&
CNTHCTL_EL2.ECV == '1' then
            if CNTPS_CTL_EL1.ENABLE == '0' then
                X[t, 64] = bits(64) UNKNOWN;
            else
                X[t, 64] = CNTPS_CVAL_EL1 -
(PhysicalCountInt() - CNTPOFF_EL2);
        else
            if CNTPS_CTL_EL1.ENABLE == '0' then
                X[t, 64] = bits(64) UNKNOWN;
            else
                X[t, 64] = CNTPS_CVAL_EL1 -
PhysicalCountInt();
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    UNDEFINED;
elsif PSTATE.EL == EL3 then
```

```
    if CNTPS_CTL_EL1.ENABLE == '0' then
        X[t, 64] = bits(64) UNKNOWN;
    else
        X[t, 64] = CNTPS_CVAL_EL1 -
PhysicalCountInt();
```

MSR CNTPS_TVAL_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b111	0b1110	0b0010	0b000

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if HaveEL(EL3) && SCR_EL3.NS == '0' then
        if SCR_EL3.EEL2 == '1' then
            UNDEFINED;
        elsif SCR_EL3.ST == '0' then
            AArch64.SystemAccessTrap(EL3, 0x18);
        elsif IsFeatureImplemented(FEAT_ECV) &&
EL2Enabled() && SCR_EL3.ECVEn == '1' &&
CNTHCTL_EL2.ECV == '1' then
            CNTPS_CVAL_EL1 = (SignExtend(X[t,
64]<31:0>, 64) + PhysicalCountInt()) - CNTPOFF_EL2;
        else
            CNTPS_CVAL_EL1 = SignExtend(X[t,
64]<31:0>, 64) + PhysicalCountInt();
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    UNDEFINED;
elsif PSTATE.EL == EL3 then
    CNTPS_CVAL_EL1 = SignExtend(X[t, 64]<31:0>, 64)
+ PhysicalCountInt();
```


28/03/2023 16:02; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94