CNTVCTSS_EL0, Counter-timer Self-Synchronized Virtual Count Register

The CNTVCTSS_EL0 characteristics are:

Purpose

Holds the 64-bit virtual count value. The virtual count value is equal to the physical count value visible in CNTPCT_EL0 minus the virtual offset visible in CNTVOFF_EL2.

Configuration

AArch64 System register CNTVCTSS_EL0 bits [63:0] are architecturally mapped to AArch32 System register CNTVCTSS[63:0].

This register is present only when FEAT_ECV is implemented. Otherwise, direct accesses to CNTVCTSS_EL0 are undefined.

All reads to the CNTVCTSS_EL0 occur in program order relative to reads to CNTVCT_EL0 or CNTVCTSS_EL0.

This register is a self-synchronised view of the CNTVCT_EL0 counter, and cannot be read speculatively.

Attributes

CNTVCTSS_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
Self-synchronized virtual count value																															
Self-synchronized virtual count value																															
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 [63:0]

Self-synchronized virtual count value.

The reset behavior of this field is:

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

Accessing CNTVCTSS_EL0

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

MRS <Xt>, CNTVCTSS_EL0

op0	op1	CRn	CRm	op2
0b11	0b011	0b1110	0b0000	0b110

```
if PSTATE.EL == EL0 then
    if !(EL2Enabled() && HCR_EL2.<E2H,TGE> == '11')
&& CNTKCTL_EL1.EL0VCTEN == '0' then
        if EL2Enabled() && HCR_EL2.TGE == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        else
            AArch64.SystemAccessTrap(EL1, 0x18);
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> == '11'
&& CNTHCTL_EL2.EL0VCTEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> != '11'
&& CNTHCTL_EL2.EL1TVCT == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        if HaveEL(EL2) && (!EL2Enabled() ||
HCR_EL2.<E2H,TGE> != '11') then
            X[t, 64] = PhysicalCountInt() -
CNTVOFF_EL2;
        else
            X[t, 64] = PhysicalCountInt();
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && CNTHCTL_EL2.EL1TVCT == '1'
then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        if HaveEL(EL2) then
            X[t, 64] = PhysicalCountInt() -
CNTVOFF_EL2;
        else
            X[t, 64] = PhysicalCountInt();
elsif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '0' then
        X[t, 64] = PhysicalCountInt() - CNTVOFF_EL2;
    else
        X[t, 64] = PhysicalCountInt();
elsif PSTATE.EL == EL3 then
    if HaveEL(EL2) && !ELUsingAArch32(EL2) then
        X[t, 64] = PhysicalCountInt() - CNTVOFF_EL2;
    elsif HaveEL(EL2) && ELUsingAArch32(EL2) then
        X[t, 64] = PhysicalCountInt() - CNTVOFF;
    else
        X[t, 64] = PhysicalCountInt();
```

28/03/2023 16:02; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94