PMINTENCLR_EL1, Performance Monitors Interrupt Enable Clear Register

The PMINTENCLR_EL1 characteristics are:

Purpose

Disables the generation of interrupt requests or, when FEAT_EBEP is implemented, PMU exceptions on overflows from the instruction counter, PMICNTR_EL0, the cycle counter, PMCCNTR_EL0, and the event counters PMEVCNTR<n>_EL0. Reading the register shows which overflow interrupt requests or PMU exceptions are enabled.

Configuration

AArch64 System register PMINTENCLR_EL1 bits [31:0] are architecturally mapped to AArch32 System register PMINTENCLR[31:0].

AArch64 System register PMINTENCLR_EL1 bits [31:0] are architecturally mapped to External register PMU.PMINTENCLR_EL1[31:0].

AArch64 System register PMINTENCLR_EL1 bits [63:32] are architecturally mapped to External register PMU.PMINTENCLR_EL1[63:32] when FEAT_PMUv3p9 is implemented or FEAT_PMUv3_EXT64 is implemented.

This register is present only when FEAT_PMUv3 is implemented. Otherwise, direct accesses to PMINTENCLR_EL1 are undefined.

Attributes

PMINTENCLR_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
RES0																											
C	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4
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

Bits [63:33]

Reserved, res0.

F<m>, bit [m+32], for m = 0
When FEAT_PMUv3_ICNTR is implemented:

Interrupt request or PMU exception on unsigned overflow of fixed-function counter <m> disable. On writes, allows software to disable the interrupt request or PMU exception on unsigned overflow of fixed-function counter <m>. On reads, returns the interrupt request or PMU exception on unsigned overflow of fixed-function counter <m> enable status.

F<m>	Meaning
`0b0`	Interrupt request or PMU exception on unsigned overflow of fixed-function counter <m> disabled.
`0b1`	Interrupt request or PMU exception on unsigned overflow of fixed-function counter <m> enabled.

PMINTENCLR_EL1.F0 holds the enable status for PMICNTR_EL0.

Accessing this field has the following behavior:

- This field reads-as-zero if all of the following are true:
    - Any of the following are true:
        - EL3 is implemented and SCR_EL3.FGTEn2 == 0.
        - HDFGRTR2_EL2.nPMICFILTR_EL0 == 0.
    - FEAT_FGT2 is implemented.
    - EL2 is implemented and enabled in the current Security state.
    - Accessed at EL1.
- This field reads-as-zero and ignores writes if any of the following are true:
    - All of the following are true:
        - EL3 is implemented.
        - MDCR_EL3.EnPM2 == 0.
        - Accessed at EL2 or EL1.
- This field ignores writes if any of the following are true:
    - All of the following are true:
        - EL3 is implemented and SCR_EL3.FGTEn2 == 0, or HDFGWTR2_EL2.nPMICFILTR_EL0 == 0.
        - FEAT_FGT2 is implemented.
        - EL2 is implemented and enabled in the current Security state.
        - Accessed at EL1.
- Otherwise access to this field is W1C.

The reset behavior of this field is:

- On a Warm reset, this field resets to `0`.

Otherwise:

Reserved, res0.

C, bit [31]

Interrupt request or PMU exception on unsigned overflow of PMCCNTR_EL0 disable. On writes, allows software to disable the interrupt request or PMU exception on unsigned overflow of PMCCNTR_EL0. On reads, returns the interrupt request or PMU exception on unsigned overflow of PMCCNTR_EL0 enable status.

C	Meaning
`0b0`	Interrupt request or PMU exception on unsigned overflow of PMCCNTR_EL0 disabled.
`0b1`	Interrupt request or PMU exception on unsigned overflow of PMCCNTR_EL0 enabled.

Access to this field is W1C.

The reset behavior of this field is:

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

P<m>, bit [m], for m = 30 to 0

Interrupt request or PMU exception on unsigned overflow of PMEVCNTR<m>_EL0 disable. On writes, allows software to disable the interrupt request or PMU exception on unsigned overflow of PMEVCNTR<m>_EL0. On reads, returns the interrupt request or PMU exception on unsigned overflow of PMEVCNTR<m>_EL0 enable status.

P<m>	Meaning
`0b0`	Interrupt request or PMU exception on unsigned overflow of PMEVCNTR<m>_EL0 disabled.
`0b1`	Interrupt request or PMU exception on unsigned overflow of PMEVCNTR<m>_EL0 enabled.

Accessing this field has the following behavior:

- This field reads-as-zero and ignores writes if any of the following are true:
    - All of the following are true:
        - EL2 is implemented and enabled in the current Security state.
        - m >= UInt(MDCR_EL2.HPMN).
        - Accessed at EL1.
    - m >= UInt(PMCR_EL0.N).
- Otherwise access to this field is W1C.

The reset behavior of this field is:

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

Accessing PMINTENCLR_EL1

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

MRS <Xt>, PMINTENCLR_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b1001	0b1110	0b010

```
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'" && MDCR_EL3.TPM == '1' then
        UNDEFINED;
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
SCR_EL3.FGTEn == '1') && HDFGRTR_EL2.PMINTEN == '1'
then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && MDCR_EL2.TPM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif HaveEL(EL3) && MDCR_EL3.TPM == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = PMINTENCLR_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
```

```
when SDD == '1'" && MDCR_EL3.TPM == '1' then
        UNDEFINED;
    elsif HaveEL(EL3) && MDCR_EL3.TPM == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = PMINTENCLR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = PMINTENCLR_EL1;
```

MSR PMINTENCLR_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1001	0b1110	0b010

```
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'" && MDCR_EL3.TPM == '1' then
        UNDEFINED;
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
SCR_EL3.FGTEn == '1') && HDFGWTR_EL2.PMINTEN == '1'
then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && MDCR_EL2.TPM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif HaveEL(EL3) && MDCR_EL3.TPM == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        PMINTENCLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && MDCR_EL3.TPM == '1' then
        UNDEFINED;
    elsif HaveEL(EL3) && MDCR_EL3.TPM == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        PMINTENCLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    PMINTENCLR_EL1 = X[t, 64];
```