PMICFILTR_EL0, Performance Monitors Instruction Counter Filter Register

The PMICFILTR_EL0 characteristics are:

Purpose

Configures the Instruction Counter.

Configuration

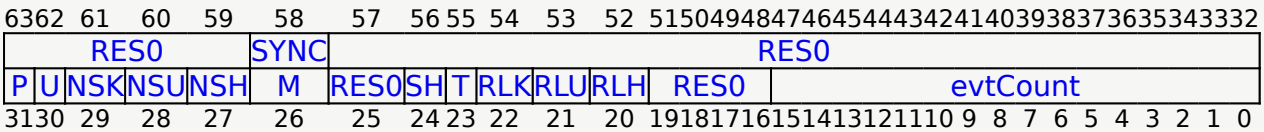
AArch64 System register PMICFILTR_EL0 bits [63:0] are architecturally mapped to External register PMU.PMICFILTR_EL0[63:0].

This register is present only when FEAT_PMUv3_ICNTR is implemented. Otherwise, direct accesses to PMICFILTR_EL0 are undefined.

Attributes

PMICFILTR_EL0 is a 64-bit register.

Field descriptions



Bits [63:59]

Reserved, res0.

SYNC, bit [58]
When FEAT_SEBEP is implemented:

Synchronous mode. Controls whether a PMU exception generated by the counter is synchronous or asynchronous.

SYNC	Meaning
0b0	Asynchronous PMU exception is enabled.
0b1	Synchronous PMU exception is enabled.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

Bits [57:32]

Reserved, res0.

P, bit [31]

EL1 filtering. Controls counting instructions in EL1.

P	Meaning
0b0	This field has no effect on filtering of instructions.
0b1	Instructions in EL1 are not counted.

If Secure and Non-secure states are implemented, then counting instructions in Non-secure EL1 is further controlled by PMICFILTR_EL0.NSK.

If FEAT_RME is implemented, then counting instructions in Realm EL1 is further controlled by PMICFILTR_EL0.RLK.

If EL3 is implemented, then counting instructions in EL3 is further controlled by PMICFILTR_EL0.M.

The reset behavior of this field is:

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

U, bit [30]

EL0 filtering. Controls counting instructions in EL0.

U	Meaning
0b0	This field has no effect on filtering of instructions.
0b1	Instructions in EL0 are not counted.

If Secure and Non-secure states are implemented, then counting instructions in Non-secure EL0 is further controlled by PMICFILTR_EL0.NSU.

If FEAT_RME is implemented, then counting instructions in Realm EL0 is further controlled by PMICFILTR_EL0.RLU.

The reset behavior of this field is:

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

NSK, bit [29]
When EL3 is implemented:

Non-secure EL1 filtering. Controls counting instructions in Non-secure EL1. If PMICFILTR_EL0.NSK is not equal to PMICFILTR_EL0.P, then instructions in Non-secure EL1 are not counted. Otherwise, PMICFILTR_EL0.NSK has no effect on filtering of instructions in Non-secure EL1.

NSK	Meaning
0b0	When PMICFILTR_EL0.P == 0, this field has no effect on filtering of instructions. When PMICFILTR_EL0.P == 1, instructions in Non-secure EL1 are not counted.
0b1	When PMICFILTR_EL0.P == 0, instructions in Non-secure EL1 are not counted. When PMICFILTR_EL0.P == 1, this field has no effect on filtering of instructions.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

NSU, bit [28]
When EL3 is implemented:

Non-secure EL0 filtering. Controls counting instructions in Non-secure EL0. If PMICFILTR_EL0.NSU is not equal to PMICFILTR_EL0.U, then instructions in Non-secure EL0 are not counted. Otherwise, PMICFILTR_EL0.NSU has no effect on filtering of instructions in Non-secure EL0.

NSU	Meaning

0b0	When PMICFILTR_EL0.U == 0, this field has no effect on filtering of instructions. When PMICFILTR_EL0.U == 1, instructions in Non-secure EL0 are not counted.
0b1	When PMICFILTR_EL0.U == 0, instructions in Non-secure EL0 are not counted. When PMICFILTR_EL0.U == 1, this field has no effect on filtering of instructions.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

NSH, bit [27]
When EL2 is implemented:

EL2 filtering. Controls counting instructions in EL2.

NSH	Meaning
0b0	Instructions in EL2 are not counted.
0b1	This field has no effect on filtering of instructions.

If EL3 is implemented and FEAT_SEL2 is implemented, then counting instructions in Secure EL2 is further controlled by PMICFILTR_EL0.SH.

If FEAT_RME is implemented, then counting instructions in Realm EL2 is further controlled by PMICFILTR_EL0.RLH.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

M, bit [26]
When EL3 is implemented:

EL3 filtering. Controls counting instructions in EL3. If PMICFILTR_EL0.M is not equal to PMICFILTR_EL0.P, then instructions in EL3 are not counted. Otherwise, PMICFILTR_EL0.M has no effect on filtering of instructions in EL3.

M	Meaning
0b0	When PMICFILTR_EL0.P == 0, this field has no effect on filtering of instructions. When PMICFILTR_EL0.P == 1, instructions in EL3 are not counted.
0b1	When PMICFILTR_EL0.P == 0, instructions in EL3 are not counted. When PMICFILTR_EL0.P == 1, this field has no effect on filtering of instructions.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

Bit [25]

Reserved, res0.

SH, bit [24]
When EL3 is implemented and FEAT_SEL2 is implemented:

Secure EL2 filtering. Controls counting instructions in Secure EL2. If PMICFILTR_EL0.SH is equal to PMICFILTR_EL0.NSH, then instructions in Secure EL2 are not counted. Otherwise, PMICFILTR_EL0.SH has no effect on filtering of instructions in Secure EL2.

SH	Meaning

0b0	When PMICFILTR_EL0.NSH == 0, instructions in Secure EL2 are not counted. When PMICFILTR_EL0.NSH == 1, this field has no effect on filtering of instructions.
0b1	When PMICFILTR_EL0.NSH == 0, this field has no effect on filtering of instructions. When PMICFILTR_EL0.NSH == 1, instructions in Secure EL2 are not counted.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

T, bit [23]
When FEAT_TME is implemented:

Non-transactional state filtering. Controls counting instructions in Non-transactional state.

T	Meaning
0b0	This field has no effect on filtering of instructions.
0b1	Instructions in Non-transactional state are not counted.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

RLK, bit [22]
When FEAT_RME is implemented:

Realm EL1 filtering. Controls counting instructions in Realm EL1. If PMICFILTR_EL0.RLK is not equal to PMICFILTR_EL0.P, then instructions in Realm EL1 are not counted. Otherwise, PMICFILTR_EL0.RLK has no effect on filtering of instructions in Realm EL1.

RLK	Meaning
0b0	When PMICFILTR_EL0.P == 0, this field has no effect on filtering of instructions. When PMICFILTR_EL0.P == 1, instructions in Realm EL1 are not counted.
0b1	When PMICFILTR_EL0.P == 0, instructions in Realm EL1 are not counted. When PMICFILTR_EL0.P == 1, this field has no effect on filtering of instructions.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

RLU, bit [21]
When FEAT_RME is implemented:

Realm EL0 filtering. Controls counting instructions in Realm EL0. If PMICFILTR_EL0.RLU is not equal to PMICFILTR_EL0.U, then instructions in Realm EL0 are not counted. Otherwise, PMICFILTR_EL0.RLU has no effect on filtering of instructions in Realm EL0.

RLU	Meaning
0b0	When PMICFILTR_EL0.U == 0, this field has no effect on filtering of instructions. When PMICFILTR_EL0.U == 1, instructions in Realm EL0 are not counted.

0b1	When PMICFILTR_EL0.U == 0, instructions in Realm EL0 are not counted. When PMICFILTR_EL0.U == 1, this field has no effect on filtering of instructions.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

RLH, bit [20]
When FEAT_RME is implemented:

Realm EL2 filtering. Controls counting instructions in Realm EL2. If PMICFILTR_EL0.RLH is equal to PMICFILTR_EL0.NSH, then instructions in Realm EL2 are not counted. Otherwise, PMICFILTR_EL0.RLH has no effect on filtering of instructions in Realm EL2.

RLH	Meaning
0b0	When PMICFILTR_EL0.NSH == 0, instructions in Realm EL2 are not counted. When PMICFILTR_EL0.NSH == 1, this field has no effect on filtering of instructions.
0b1	When PMICFILTR_EL0.NSH == 0, this field has no effect on filtering of instructions. When PMICFILTR_EL0.NSH == 1, instructions in Realm EL2 are not counted.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

Bits [19:16]

Reserved, res0.

evtCount, bits [15:0]

Event to count.

Reads as `0x0008`.

Access to this field is **RO**.

Accessing PMICFILTR_EL0

PMICFILTR_EL0 reads-as-zero and ignores writes if all of the following are true:

- PSTATE.EL == EL0.
- PMUACR_EL1.F0 == 0.

PMICFILTR_EL0 ignores writes if all of the following are true:

- PSTATE.EL == EL0.
- PMUSERENR_EL0.IR == 1.

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

MRS <Xt>, PMICFILTR_EL0

op0	op1	CRn	CRm	op2
0b11	0b011	0b1001	0b0110	0b000

```
if PSTATE.EL == EL0 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && MDCR_EL3.EnPM2 == '0' then
        UNDEFINED;
    elsif Halted() && HaveEL(EL3) && EDSCR.SDD ==
'1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && MDCR_EL3.TPM == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> != '11'
&& IsFeatureImplemented(FEAT_FGT2) && HaveEL(EL3) &&
SCR_EL3.FGTEn2 == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> != '11'
&& IsFeatureImplemented(FEAT_FGT2) &&
HDFGRTR2_EL2.nPMICFILTR_EL0 == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
```

```
    elsif EL2Enabled() && MDCR_EL2.TPM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif HaveEL(EL3) && MDCR_EL3.EnPM2 == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 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] = PMICFILTR_EL0;
elsif PSTATE.EL == EL1 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && MDCR_EL3.EnPM2 == '0' then
        UNDEFINED;
    elsif 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_FGT2) && HaveEL(EL3) &&
SCR_EL3.FGTEn2 == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT2) &&
HDFGRTR2_EL2.nPMICFILTR_EL0 == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && MDCR_EL2.TPM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif HaveEL(EL3) && MDCR_EL3.EnPM2 == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 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] = PMICFILTR_EL0;
elsif PSTATE.EL == EL2 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && MDCR_EL3.EnPM2 == '0' then
        UNDEFINED;
    elsif 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.EnPM2 == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 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] = PMICFILTR_EL0;
elsif PSTATE.EL == EL3 then
    X[t, 64] = PMICFILTR_EL0;
```

MSR PMICFILTR_EL0, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b011	0b1001	0b0110	0b000

```
if PSTATE.EL == EL0 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && MDCR_EL3.EnPM2 == '0' then
        UNDEFINED;
    elsif Halted() && HaveEL(EL3) && EDSCR.SDD ==
'1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && MDCR_EL3.TPM == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> != '11'
&& IsFeatureImplemented(FEAT_FGT2) && HaveEL(EL3) &&
SCR_EL3.FGTEn2 == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<E2H,TGE> != '11'
&& IsFeatureImplemented(FEAT_FGT2) &&
HDFGWTR2_EL2.nPMICFILTR_EL0 == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && MDCR_EL2.TPM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif HaveEL(EL3) && MDCR_EL3.EnPM2 == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    elsif HaveEL(EL3) && MDCR_EL3.TPM == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        PMICFILTR_EL0 = X[t, 64];
elsif PSTATE.EL == EL1 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && MDCR_EL3.EnPM2 == '0' then
        UNDEFINED;
    elsif 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_FGT2) && HaveEL(EL3) &&
    SCR_EL3.FGTEn2 == '0' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        elsif EL2Enabled() &&
    IsFeatureImplemented(FEAT_FGT2) &&
    HDFGWTR2_EL2.nPMICFILTR_EL0 == '0' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        elsif EL2Enabled() && MDCR_EL2.TPM == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        elsif HaveEL(EL3) && MDCR_EL3.EnPM2 == '0' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
        elsif HaveEL(EL3) && MDCR_EL3.TPM == '1' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
        else
            PMICFILTR_EL0 = 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.EnPM2 == '0' then
            UNDEFINED;
        elsif 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.EnPM2 == '0' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
        elsif HaveEL(EL3) && MDCR_EL3.TPM == '1' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
        else
            PMICFILTR_EL0 = X[t, 64];
    elsif PSTATE.EL == EL3 then
        PMICFILTR_EL0 = X[t, 64];
```


28/03/2023 16:01; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94