PFAR_EL1, Physical Fault Address Register (EL1)

The PFAR_EL1 characteristics are:

Purpose

Records the faulting physical address for a synchronous External Abort, or SError exception taken to EL1.

Configuration

This register is present only when FEAT_PFAR is implemented. Otherwise, direct accesses to PFAR_EL1 are undefined.

Attributes

PFAR_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
NS	NSE	RES0	PA[55:52]	PA[51:48]	PA
PA					
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

NS, bit [63]
When FEAT_RME is implemented:

Together with PFAR_EL1.NSE, reports the physical address space of the access that triggered the exception.

NSE	NS	Meaning
`0b0`	`0b0`	When Secure state is implemented, Secure. Otherwise reserved.
`0b0`	`0b1`	Non-secure.
`0b1`	`0b0`	Reserved.
`0b1`	`0b1`	Realm.

The reset behavior of this field is:

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

When EL3 is implemented:

Non-secure. Reports the physical address space of the access that triggered the exception.

NS	Meaning
`0b0`	Secure physical address space.
`0b1`	Non-secure physical address space.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

NSE, bit [62]
When FEAT_RME is implemented:

Together with PFAR_EL1.NS, reports the physical address space of the access that triggered the exception.

For a description of the values derived by evaluating NS and NSE together, see MFAR_EL3.NS.

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

Bits [61:56]

Reserved, res0.

PA[55:52], bits [55:52]
When FEAT_D128 is implemented:

When FEAT_D128 is implemented, extension to PFAR_EL1.PA[47:0].

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

PA[51:48], bits [51:48]
When FEAT_LPA is implemented:

When FEAT_LPA is implemented, extension to PFAR_EL1.PA[47:0].

The reset behavior of this field is:

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

Otherwise:

Reserved, res0.

PA, bits [47:0]

Physical Address. Bits [47:0] of the aborting physical address.

For implementations with fewer than 48 physical address bits, the corresponding upper bits in this field are res0.

The recorded address can be any address within the same naturally-aligned fault granule as the faulting physical address, where the size of the fault granule is implementation defined and no larger than the larger than:

- The size of the range of values permitted to be recorded in FAR_EL1.

The reset behavior of this field is:

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

Accessing PFAR_EL1

PFAR_EL1 is not valid and reads unknown if ESR_EL1.PFV is recorded as 0.

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

MRS <Xt>, PFAR_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0110	0b0000	0b101

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT2) && HaveEL(EL3) &&
SCR_EL3.FGTEn2 == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT2) &&
HFGRTR2_EL2.nPFAR_EL1 == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<NV2,NV1,NV> ==
'111' then
        X[t, 64] = NVMem[0x2D0];
    else
        X[t, 64] = PFAR_EL1;
elsif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' then
        X[t, 64] = PFAR_EL2;
    else
        X[t, 64] = PFAR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = PFAR_EL1;
```

MSR PFAR_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0110	0b0000	0b101

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT2) && HaveEL(EL3) &&
SCR_EL3.FGTEn2 == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT2) &&
HFGWTR2_EL2.nPFAR_EL1 == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && HCR_EL2.<NV2,NV1,NV> ==
'111' then
        NVMem[0x2D0] = X[t, 64];
    else
```

```
        PFAR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' then
        PFAR_EL2 = X[t, 64];
    else
        PFAR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    PFAR_EL1 = X[t, 64];
```

MRS <Xt>, PFAR_EL12

op0	op1	CRn	CRm	op2
0b11	0b101	0b0110	0b0000	0b101

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV1,NV> == '101'
then
        X[t, 64] = NVMem[0x2D0];
    elsif EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    if HCR_EL2.E2H == '1' then
        X[t, 64] = PFAR_EL1;
    else
        UNDEFINED;
elsif PSTATE.EL == EL3 then
    if EL2Enabled() && !ELUsingAArch32(EL2) &&
HCR_EL2.E2H == '1' then
        X[t, 64] = PFAR_EL1;
    else
        UNDEFINED;
```

MSR PFAR_EL12, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b101	0b0110	0b0000	0b101

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV1,NV> == '101'
then
```

```
            NVMem[0x2D0] = 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
            PFAR_EL1 = X[t, 64];
        else
            UNDEFINED;
    elsif PSTATE.EL == EL3 then
        if EL2Enabled() && !ELUsingAArch32(EL2) &&
    HCR_EL2.E2H == '1' then
            PFAR_EL1 = X[t, 64];
        else
            UNDEFINED;
```