ICC_SRE_EL2, Interrupt Controller System Register Enable Register (EL2)

The ICC_SRE_EL2 characteristics are:

Purpose

Controls whether the System register interface or the memory-mapped interface to the GIC CPU interface is used for EL2.

Configuration

AArch64 System register ICC_SRE_EL2 is architecturally mapped to AArch32 System register ICC_HSRE.

This register is present only when FEAT_GICv3 is implemented and (EL2 is implemented or EL3 is implemented). Otherwise, direct accesses to ICC_SRE_EL2 are undefined.

If EL2 is not implemented, this register is res0 from EL3.

This register has no effect if EL2 is not enabled in the current Security state.

Attributes

ICC_SRE_EL2 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																															
RES0																												Enable	DIB	DFB	SRE
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 [63:4]

Reserved, res0.

Enable, bit [3]

Enable. Enables lower Exception level access to ICC_SRE_EL1.

Enable	Meaning

0b0	When EL2 is implemented and enabled in the current Security state, EL1 accesses to ICC_SRE_EL1 trap to EL2.
0b1	EL1 accesses to ICC_SRE_EL1 do not trap to EL2.

If ICC_SRE_EL2.SRE is RAO/WI, an implementation is permitted to make the Enable bit RAO/WI.

If ICC_SRE_EL2.SRE is 0, the Enable bit behaves as 1 for all purposes other than reading the value of the bit.

The reset behavior of this field is:

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

DIB, bit [2]

Disable IRQ bypass.

DIB	Meaning
0b0	IRQ bypass enabled.
0b1	IRQ bypass disabled.

If EL3 is implemented and GICD_CTLR.DS is 0, this field is a read-only alias of ICC_SRE_EL3.DIB.

If EL3 is implemented and GICD_CTLR.DS is 1, this field is a read/write alias of ICC_SRE_EL3.DIB.

In systems that do not support IRQ bypass, this bit is RAO/WI.

The reset behavior of this field is:

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

DFB, bit [1]

Disable FIQ bypass.

DFB	Meaning
0b0	FIQ bypass enabled.
0b1	FIQ bypass disabled.

If EL3 is implemented and GICD_CTLR.DS is 0, this field is a read-only alias of ICC_SRE_EL3.DFB.

If EL3 is implemented and GICD_CTLR.DS is 1, this field is a read/write alias of ICC_SRE_EL3.DFB.

In systems that do not support FIQ bypass, this bit is RAO/WI.

The reset behavior of this field is:

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

SRE, bit [0]

System Register Enable.

SRE	Meaning
`0b0`	The memory-mapped interface must be used. Access at EL2 to any ICH_* or ICC_* register other than ICC_SRE_EL1 or ICC_SRE_EL2, is trapped to EL2.
`0b1`	The System register interface to the ICH_* registers and the EL1 and EL2 ICC_* registers is enabled for EL2.

If software changes this bit from 1 to 0, the results are unpredictable.

If an implementation supports only a System register interface to the GIC CPU interface, this bit is RAO/WI.

If EL3 is implemented and ICC_SRE_EL3.SRE==0 this bit is RAZ/WI. If ICC_SRE_EL3.SRE is changed from zero to one, this bit becomes unknown.

If Realm Management Extension is implemented, this field is RAO/WI.

FEAT_GICv3 implementations that do not require GICv2 compatibility might choose to make this bit RAO/WI, but this is only allowed if ICC_SRE_EL3.SRE is also RAO/WI.

The reset behavior of this field is:

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

Accessing ICC_SRE_EL2

Execution with ICC_SRE_EL2.SRE set to 0 might make some System registers unknown.

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

MRS <Xt>, ICC_SRE_EL2

op0	op1	CRn	CRm	op2
0b11	0b100	0b1100	0b1001	0b101

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && ICC_SRE_EL3.Enable == '0' then
        UNDEFINED;
    elsif HaveEL(EL3) && ICC_SRE_EL3.Enable == '0'
then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ICC_SRE_EL2;
elsif PSTATE.EL == EL3 then
    if !EL2Enabled() then
        UNDEFINED;
    else
        X[t, 64] = ICC_SRE_EL2;
```

MSR ICC_SRE_EL2, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b100	0b1100	0b1001	0b101

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.NV == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
&& boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && ICC_SRE_EL3.Enable == '0' then
        UNDEFINED;
    elsif HaveEL(EL3) && ICC_SRE_EL3.Enable == '0'
```

```
then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ICC_SRE_EL2 = X[t, 64];
elsif PSTATE.EL == EL3 then
    if !EL2Enabled() then
        UNDEFINED;
    else
        ICC_SRE_EL2 = X[t, 64];
```

28/03/2023 16:01; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94