

SCTLR2_EL1, System Control Register (EL1)

The SCTLR2_EL1 characteristics are:

Purpose

Provides top level control of the system, including its memory system, at EL1 and EL0.

Configuration

This register is present only when FEAT_SCTLR2 is implemented. Otherwise, direct accesses to SCTLR2_EL1 are undefined.

Attributes

SCTLR2_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
RES0																															
RES0																															
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
																										EnIDCP128	EASE	EnANERR	EnADERR	NMEA	RES0

Bits [63:7]

Reserved, res0.

EnIDCP128, bit [6] When FEAT_SYSREG128 is implemented:

Enables access to implementation defined 128-bit System registers.

EnIDCP128	Meaning
-----------	---------

0b0	Accesses at EL0 to implementation defined 128-bit System registers are trapped to EL1 using an ESR_EL1.EC value of 0x14, unless the access generates a higher priority exception. Disables the functionality of the 128-bit implementation defined System registers that are accessible at EL1.
0b1	No accesses are trapped by this control.

The reset behavior of this field is:

- On a Warm reset:
 - When EL2 is not implemented and EL3 is not implemented, this field resets to 0.
 - Otherwise, this field resets to an architecturally unknown value.

Otherwise:

Reserved, res0.

EASE, bit [5]

When FEAT_DoubleFault2 is implemented:

External Aborts to SError interrupt exception vector.

EASE	Meaning
0b0	Synchronous External Abort exceptions taken to EL1 are taken to the appropriate synchronous exception vector offset from VBAR_EL1 .
0b1	Synchronous External Abort exceptions taken to EL1 are taken to the appropriate SError exception vector offset from VBAR_EL1 .

This field is ignored by the PE and treated as zero when any of the following are true:

- All of the following are true:
 - EL2 is implemented and enabled in the current Security state.
 - The Effective value of [HCRX_EL2](#).SCTLR2En is 0.
- [SCR_EL3](#).SCTLR2En == 0.

The reset behavior of this field is:

- On a Warm reset:
 - When EL2 is not implemented and EL3 is not implemented, this field resets to 0.
 - Otherwise, this field resets to an architecturally unknown value.

Otherwise:

Reserved, res0.

EnANERR, bit [4]

When FEAT_ANERR is implemented:

Enable Asynchronous Normal Read Error.

EnANERR	Meaning
0b0	External abort on Normal memory reads generate synchronous Data Abort exceptions in the EL1&0 translation regime.
0b1	External abort on Normal memory reads generate synchronous Data Abort or asynchronous SError exceptions in the EL1&0 translation regime.

It is implementation-specific whether this field applies to memory reads generated by each of the following:

- FP&SIMD register loads.
- SVE register loads.
- SME register loads.
- LD<op>, SWP and CAS{P} Atomic instructions that return a value to the PE.
- ST64BV{0} instructions that return a value to the PE.
- RCW instructions that return a value to the PE.

Setting this field to 0 does not guarantee that the PE is able to take a synchronous Data Abort exception for an External abort on a Normal memory read in every case. There might be implementation-specific circumstances when an error on a load cannot be taken synchronously. These circumstances should be rare enough that treating such occurrences as fatal does not cause a significant increase in failure rate.

Setting this field to 0 might have a performance impact for Normal memory reads.

This field is ignored by the PE and treated as zero when any of the following are true:

- All of the following are true:
 - EL2 is implemented and enabled in the current Security state.
 - The Effective value of [HCRX_EL2](#).SCTLR2En is 0.
- All of the following are true:
 - EL2 is implemented and enabled in the current Security state.
 - The Effective value of [HCRX_EL2](#).EnSNERR is 1.
- [SCR_EL3](#).SCTLR2En == 0.

The reset behavior of this field is:

- On a Warm reset:
 - When EL2 is not implemented and EL3 is not implemented, this field resets to 0.
 - Otherwise, this field resets to an architecturally unknown value.

Otherwise:

Reserved, res0.

EnADERR, bit [3]

When FEAT_ADERR is implemented:

Enable Asynchronous Device Read Error.

EnADERR	Meaning
0b0	External abort on Device memory reads generate synchronous Data Abort exceptions in the EL1&0 translation regime.

0b1	External abort on Device memory reads generate synchronous Data Abort or asynchronous SError exceptions in the EL1&0 translation regime.
-----	--

It is implementation-specific whether this field applies to memory reads generated by each of the following:

- FP&SIMD register loads.
- SVE register loads.
- SME register loads.
- LD<op>, SWP and CAS{P} Atomic instructions that return a value to the PE.
- ST64BV{0} instructions that return a value to the PE.
- RCW instructions that return a value to the PE.

Setting this field to 0 does not guarantee that the PE is able to take a synchronous Data Abort exception for an External abort on a Device memory read in every case. There might be implementation-specific circumstances when an error on a load cannot be taken synchronously. These circumstances should be rare enough that treating such occurrences as fatal does not cause a significant increase in failure rate.

Setting this field to 0 might have a performance impact for Device memory reads.

This field is ignored by the PE and treated as zero when any of the following are true:

- All of the following are true:
 - EL2 is implemented and enabled in the current Security state.
 - The Effective value of [HCRX_EL2](#).SCTLR2En is 0.
- All of the following are true:
 - EL2 is implemented and enabled in the current Security state.
 - The Effective value of [HCRX_EL2](#).EnSDERR is 1.
- [SCR_EL3](#).SCTLR2En == 0.

The reset behavior of this field is:

- On a Warm reset:
 - When EL2 is not implemented and EL3 is not implemented, this field resets to 0.
 - Otherwise, this field resets to an architecturally unknown value.

Otherwise:

Reserved, res0.

NMEA, bit [2]**When FEAT_DoubleFault2 is implemented:**

Non-maskable External Aborts. Controls whether PSTATE.A masks SError exceptions at EL1.

NMEA	Meaning
0b0	SError exceptions are not taken at EL1 if PSTATE.A == 1, unless routed to a higher Exception level.
0b1	SError exceptions are taken at EL1 regardless of the value of PSTATE.A, unless routed to a higher Exception level.

This field is ignored by the PE and treated as zero when any of the following are true:

- All of the following are true:
 - EL2 is implemented and enabled in the current Security state.
 - The Effective value of [HCRX_EL2](#).SCTLR2En is 0.
- [SCR_EL3](#).SCTLR2En == 0.

The reset behavior of this field is:

- On a Warm reset:
 - When EL2 is not implemented and EL3 is not implemented, this field resets to 0.
 - Otherwise, this field resets to an architecturally unknown value.

Otherwise:

Reserved, res0.

Bits [1:0]

Reserved, res0.

Accessing SCTLR2_EL1

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

MRS <Xt>, SCTLR2_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0001	0b0000	0b011

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
    && boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && SCR_EL3.SCTLR2En == '0' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TRVM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() &&
IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
SCR_EL3.FGTEn == '1') && HFGTR_EL2.SCTLR_EL1 == '1'
then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && (!IsHCRXEL2Enabled() ||
HCRX_EL2.SCTLR2En == '0') then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif HaveEL(EL3) && SCR_EL3.SCTLR2En == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    elseif EL2Enabled() && HCR_EL2.<NV2,NV1,NV> ==
'111' then
        X[t, 64] = NVMem[0x278];
    else
        X[t, 64] = SCTLR2_EL1;
elseif PSTATE.EL == EL2 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
    && boolean IMPLEMENTATION_DEFINED "EL3 trap priority
when SDD == '1'" && SCR_EL3.SCTLR2En == '0' then
        UNDEFINED;
    elseif HaveEL(EL3) && SCR_EL3.SCTLR2En == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    elseif HCR_EL2.E2H == '1' then
        X[t, 64] = SCTLR2_EL2;
    else
        X[t, 64] = SCTLR2_EL1;
elseif PSTATE.EL == EL3 then
    X[t, 64] = SCTLR2_EL1;

```

MSR SCTLR2_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0001	0b0000	0b011

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
    && boolean IMPLEMENTATION_DEFINED "EL3 trap priority
    when SDD == '1'" && SCR_EL3.SCTLR2En == '0' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TVM == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() &&
    IsFeatureImplemented(FEAT_FGT) && (!HaveEL(EL3) ||
    SCR_EL3.FGTEn == '1') && HFGWTR_EL2.SCTLR_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && (!IsHCRXEL2Enabled() ||
    HCRX_EL2.SCTLR2En == '0') then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif HaveEL(EL3) && SCR_EL3.SCTLR2En == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    elseif EL2Enabled() && HCR_EL2.<NV2,NV1,NV> ==
    '111' then
        NVMem[0x278] = X[t, 64];
    else
        SCTLR2_EL1 = X[t, 64];
elseif PSTATE.EL == EL2 then
    if Halted() && HaveEL(EL3) && EDSCR.SDD == '1'
    && boolean IMPLEMENTATION_DEFINED "EL3 trap priority
    when SDD == '1'" && SCR_EL3.SCTLR2En == '0' then
        UNDEFINED;
    elseif HaveEL(EL3) && SCR_EL3.SCTLR2En == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    elseif HCR_EL2.E2H == '1' then
        SCTLR2_EL2 = X[t, 64];
    else
        SCTLR2_EL1 = X[t, 64];
elseif PSTATE.EL == EL3 then
    SCTLR2_EL1 = X[t, 64];
```


MRS <Xt>, SCTLR2_EL12

op0	op1	CRn	CRm	op2
0b11	0b101	0b0001	0b0000	0b011

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.<NV2,NV1,NV> == '101'
    then
        X[t, 64] = NVMem[0x278];
        elsif EL2Enabled() && HCR_EL2.NV == '1' then
            AArch64.SystemAccessTrap(EL2, 0x18);
        else
            UNDEFINED;
    elsif PSTATE.EL == EL2 then
        if HCR_EL2.E2H == '1' then
            if Halted() && HaveEL(EL3) && EDSCR.SDD ==
            '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
            priority when SDD == '1'" && SCR_EL3.SCTLR2En == '0'
            then
                UNDEFINED;
            elsif HaveEL(EL3) && SCR_EL3.SCTLR2En == '0'
            then
                if Halted() && EDSCR.SDD == '1' then
                    UNDEFINED;
                else
                    AArch64.SystemAccessTrap(EL3, 0x18);
            else
                X[t, 64] = SCTLR2_EL1;
            else
                UNDEFINED;
        elsif PSTATE.EL == EL3 then
            if EL2Enabled() && !ELUsingAArch32(EL2) &&
            HCR_EL2.E2H == '1' then
                X[t, 64] = SCTLR2_EL1;
            else
                UNDEFINED;
```

MSR SCTLR2_EL12, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b101	0b0001	0b0000	0b011

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

```

        NVMem[0x278] = 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
            if Halted() && HaveEL(EL3) && EDSCR.SDD ==
'1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && SCR_EL3.SCTLR2En == '0'
then
                UNDEFINED;
            elsif HaveEL(EL3) && SCR_EL3.SCTLR2En == '0'
then
                if Halted() && EDSCR.SDD == '1' then
                    UNDEFINED;
                else
                    AArch64.SystemAccessTrap(EL3, 0x18);
                else
                    SCTLR2_EL1 = X[t, 64];
            else
                UNDEFINED;
        elsif PSTATE.EL == EL3 then
            if EL2Enabled() && !ELUsingAArch32(EL2) &&
HCR_EL2.E2H == '1' then
                SCTLR2_EL1 = X[t, 64];
            else
                UNDEFINED;

```

[AArch32
Registers](#)

[AArch64
Registers](#)

[AArch32
Instructions](#)

[AArch64
Instructions](#)

[Index by
Encoding](#)

[External
Registers](#)

28/03/2023 16:02; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94

Copyright © 2010-2023 Arm Limited or its affiliates. All rights reserved. This document is Non-Confidential.