

RMR_EL1, Reset Management Register (EL1)

The RMR_EL1 characteristics are:

Purpose

When this register is implemented:

- A write to the register at EL1 can request a Warm reset.
- If EL1 can use all Execution states, this register specifies the Execution state that the PE boots into on a Warm reset.

Configuration

AArch64 System register RMR_EL1 bits [31:0] are architecturally mapped to AArch32 System register [RMR\[31:0\]](#) when the highest implemented Exception level is EL1.

This register is present only when the highest implemented Exception level is EL1. Otherwise, direct accesses to RMR_EL1 are undefined.

When EL1 is the highest implemented Exception level:

- If EL1 can use all Execution states then this register must be implemented.
- If EL1 cannot use AArch32 then it is implementation defined whether the register is implemented.

Attributes

RMR_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																															RR	AA64
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 [63:2]

Reserved, res0.

RR, bit [1]

Reset Request. Setting this bit to 1 requests a Warm reset.

The reset behavior of this field is:

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

AA64, bit [0]

When EL1 is capable of using AArch32:

When EL1 can use AArch32, determines which Execution state the PE boots into after a Warm reset:

AA64	Meaning
0b0	AArch32.
0b1	AArch64.

On coming out of the Warm reset, execution starts at the implementation defined reset vector address of the specified Execution state.

If EL1 can only use AArch64 state, this bit is RAO/WI.

When implemented as a RW field, this field resets to 1 on a Cold reset.

Otherwise:

Reserved, RAO/WI.

Accessing RMR_EL1

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

MRS <Xt>, RMR_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b1100	0b0000	0b010

```
if PSTATE.EL == EL1 && IsHighestEL(EL1) then
    X[t, 64] = RMR_EL1;
else
    UNDEFINED;
```

MSR RMR_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1100	0b0000	0b010

```
if PSTATE.EL == EL1 && IsHighestEL(EL1) then
    RMR_EL1 = X[t, 64];
else
    UNDEFINED;
```

[AArch32
Registers](#)

[AArch64
Registers](#)

[AArch32
Instructions](#)

[AArch64
Instructions](#)

[Index by
Encoding](#)

[External
Registers](#)

28/03/2023 16:01; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94

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