ERRIRQSR, Error Interrupt Status Register

The ERRIRQSR characteristics are:

Purpose

Interrupt status register.

Configuration

This register is present only when interrupt configuration registers are implemented. Otherwise, direct accesses to ERRIRQSR are res0.

ERRIRQSR is implemented only as part of a memory-mapped group of error records.

Attributes

ERRIRQSR is a 64-bit register.

Field descriptions

When the implementation uses the recommended layout for the ERRIRQCR registers and the implementation uses simple interrupts:

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

Bits [63:0]

Reserved, res0.

To determine whether an interrupt is active, software must examine the individual ERR<n>STATUS registers.

When the implementation uses message-signaled interrupts and the implementation uses the recommended layout for the ERRIRQCR registers:

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																										CRIERR	CRI	ERIERR	ERI	FHIERR	FHI
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 [63:6]

Reserved, res0.

CRIERR, bit [5]
When the Critical Error Interrupt is implemented:

Critical Error Interrupt Error.

CRIERR	Meaning
0b0	Critical Error Interrupt write has not returned an error since this field was last cleared to zero.
0b1	Critical Error Interrupt write has returned an error since this field was last cleared to zero.

The reset behavior of this field is:

- On an Error recovery reset, this field resets to an architecturally unknown value.

Access to this field is **W1C**.

Otherwise:

Reserved, res0.

CRI, bit [4]
When the Critical Error Interrupt is implemented:

Critical Error Interrupt write in progress.

CRI	Meaning
0b0	Critical Error Interrupt write not in progress.
0b1	Critical Error Interrupt write in progress.

Software must not disable an interrupt whilst the write is in progress.

Note

This field does not indicate whether an interrupt is active, but rather whether a write triggered by the interrupt is in progress.

To determine whether an interrupt is active, software must examine the individual ERR<n>STATUS registers.

Access to this field is **RO**.

Otherwise:

Reserved, res0.

ERIERR, bit [3]
When the Error Recovery Interrupt is implemented:

Error Recovery Interrupt Error.

ERIERR	Meaning
0b0	Error Recovery Interrupt write has not returned an error since this field was last cleared to zero.
0b1	Error Recovery Interrupt write has returned an error since this field was last cleared to zero.

The reset behavior of this field is:

- On an Error recovery reset, this field resets to an architecturally unknown value.

Access to this field is **W1C**.

Otherwise:

Reserved, res0.

ERI, bit [2]
When the Error Recovery Interrupt is implemented:

Error Recovery Interrupt write in progress.

ERI	Meaning
0b0	Error Recovery Interrupt write not in progress.
0b1	Error Recovery Interrupt write in progress.

Software must not disable an interrupt whilst the write is in progress.

Note

This field does not indicate whether an interrupt is active, but rather whether a write triggered by the interrupt is in progress.

To determine whether an interrupt is active, software must examine the individual ERR<n>STATUS registers.

Access to this field is **RO**.

Otherwise:

Reserved, res0.

FHIERR, bit [1]
When the Fault Handling Interrupt is implemented:

Fault Handling Interrupt Error.

FHIERR	Meaning
0b0	Fault Handling Interrupt write has not returned an error since this field was last cleared to zero.
0b1	Fault Handling Interrupt write has returned an error since this field was last cleared to zero.

The reset behavior of this field is:

- On an Error recovery reset, this field resets to an architecturally unknown value.

Access to this field is **W1C**.

Otherwise:

Reserved, res0.

FHI, bit [0]
When the Fault Handling Interrupt is implemented:

Fault Handling Interrupt write in progress.

FHI	Meaning

0b0	Fault Handling Interrupt write not in progress.
0b1	Fault Handling Interrupt write in progress.

Software must not disable an interrupt whilst the write is in progress.

Note

This field does not indicate whether an interrupt is active, but rather whether a write triggered by the interrupt is in progress.

To determine whether an interrupt is active, software must examine the individual ERR<n>STATUS registers.

Access to this field is **RO**.

Otherwise:

Reserved, res0.

When the implementation does not use the recommended layout for the ERRIRQCR registers:

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
IMPLEMENTATION DEFINED																															
IMPLEMENTATION DEFINED																															
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

IMPLEMENTATION DEFINED, bits [63:0]

implementation defined.

Accessing ERRIRQSR

If the implementation does not use the recommended layout for the ERRIRQCR registers then accesses to ERRIRQSR are implementation defined.

ERRIRQSR can be accessed through the memory-mapped interfaces:

Component	Offset	Instance
RAS	0xEF8	ERRIRQSR

This interface is accessible as follows:

- When the implementation uses message-signaled interrupts, (an access is Non-secure or an access is Realm), the implementation uses the recommended layout for the ERRIRQCR registers and ERRIRQSR.NSMSI configures the physical address space for message-signaled interrupts as Secure, accesses to this register are **RO**.
- Otherwise, accesses to this register are **RW**.

28/03/2023 16:02; 72747e43966d6b97dcbd230a1b3f0421d1ea3d94