MSMON_CFG_MON_SEL, MPAM Monitor Instance Selection Register

The MSMON_CFG_MON_SEL characteristics are:

Purpose

Selects a monitor instance to access through the MSMON configuration and counter registers.

MSMON_CFG_MON_SEL_s selects a Secure monitor instance to access via the Secure MPAM feature page. MSMON_CFG_MON_SEL_ns selects a Non-secure monitor instance to access via the Non-secure MPAM feature page. MSMON_CFG_MON_SEL_rt selects a Root monitor instance to access via the Root MPAM feature page. MSMON_CFG_MON_SEL_rl selects a Realm monitor instance to access via the Realm MPAM feature page.

Note

Different performance monitoring features within an MSC could have different numbers of monitor instances. See the NUM_MON field in the corresponding ID register. This means that a monitor out-of-bounds error might be signaled when an MSMON_CFG register is accessed because the value in MSMON_CFG_MON_SEL.MON_SEL is too large for the particular monitoring feature.

To configure a monitor, set MON_SEL in this register to the index of the monitor instance to configure, then write to the MSMON_CFG_x register to set the configuration of the monitor. At a later time, read the monitor register (for example, MSMON_MBWU) to get the value of the monitor.

Configuration

This register is present only when FEAT_MPAM is implemented and (MPAMF_IDR.HAS_MSMON == 1, or (MPAMF_IDR.HAS_IMPL_IDR == 1 and MPAMF_IDR.EXT == 0) or (MPAMF_IDR.HAS_IMPL_IDR == 1, MPAMF_IDR.EXT == 1 and MPAMF_IDR.NO_IMPL_MSMON == 0)). Otherwise, direct accesses to MSMON_CFG_MON_SEL are res0.

The power and reset domain of each MSC component is specific to that component.

Attributes

MSMON_CFG_MON_SEL is a 32-bit register.

Field descriptions

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
RES0	RIS	RES0	MON_SEL

Bits [31:28]

Reserved, res0.

RIS, bits [27:24]
When (FEAT_MPAMv0p1 is implemented or FEAT_MPAMv1p1 is implemented), MPAMF_IDR.EXT == 1 and MPAMF_IDR.HAS_RIS == 1:

Resource Instance Selector. RIS selects one resource to configure through MSMON_CFG registers.

Otherwise:

Reserved, res0.

Bits [23:16]

Reserved, res0.

MON_SEL, bits [15:0]

Selects the monitor instance to configure or read.

Reads and writes to other MSMON registers are indexed by MON_SEL and by the NS bit used to access MSMON_CFG_MON_SEL to access the configuration for a single monitor.

Accessing MSMON_CFG_MON_SEL

This register is within the MPAM feature page memory frames.

In a system that supports Secure, Non-secure, Root, and Realm memory maps, there must be MPAM feature pages in all four address maps:

- MSMON_CFG_MON_SEL_s must only be accessible from the Secure MPAM feature page.
- MSMON_CFG_MON_SEL_ns must only be accessible from the Non-secure MPAM feature page.

- MSMON_CFG_MON_SEL_rt must only be accessible from the Root MPAM feature page.
- MSMON_CFG_MON_SEL_rl must only be accessible from the Realm MPAM feature page.

MSMON_CFG_MON_SEL_s, MSMON_CFG_MON_SEL_ns, MSMON_CFG_MON_SEL_rt, and MSMON_CFG_MON_SEL_rl must be separate registers:

- The Secure instance (MSMON_CFG_MON_SEL_s) accesses the monitor instance selector used for Secure PARTIDs.
- The Non-secure instance (MSMON_CFG_MON_SEL_ns) accesses the monitor instance selector used for Non-secure PARTIDs.
- The Root instance (MSMON_CFG_MON_SEL_rt) accesses the monitor instance selector used for Root PARTIDs.
- The Realm instance (MSMON_CFG_MON_SEL_rl) accesses the monitor instance selector used for Realm PARTIDs.

MSMON_CFG_MON_SEL can be accessed through the memory-mapped interfaces:

Component	Frame	Offset	Instance
MPAM	MPAMF_BASE_s	0x0800	MSMON_CFG_MON_SEL_s

Accesses on this interface are **RW**.

Component	Frame	Offset	Instance
MPAM	MPAMF_BASE_ns	0x0800	MSMON_CFG_MON_SEL_ns

Accesses on this interface are **RW**.

Component	Frame	Offset	Instance
MPAM	MPAMF_BASE_rt	0x0800	MSMON_CFG_MON_SEL_rt

When FEAT_RME is implemented, accesses on this interface are **RW**.

Component	Frame	Offset	Instance
MPAM	MPAMF_BASE_rl	0x0800	MSMON_CFG_MON_SEL_rl

When FEAT_RME is implemented, accesses on this interface are **RW**.